
Mobile Doctor's Desktop

Service de sécurité et de mise à jour pour clients mobiles

Projet de diplôme

Réalisé par : Stephen Badan
Date : Septembre à décembre 2007
Classe : TR 2007
Section : Télécommunications, réseaux et services

Professeur responsable : Markus Jatton



Imprimé le 12 décembre 2007

1. Remerciements

La réalisation d'un travail de diplôme ne peut se faire sans aide ou conseils venant de l'extérieur, c'est pourquoi je tiens à remercier les personnes m'ayant soutenues durant ces trois mois de labeur.

Je souhaite tout d'abord adresser un très grand merci à Monsieur Jatton qui m'a guidé tout au long du projet et qui m'a donné de précieux conseils et idées sans lesquels la réalisation du projet aurait été bien plus difficile. Je remercie aussi Christophe Greppin pour les diverses solutions qu'il m'a conseillées.

Je souhaite par ailleurs remercier toutes les personnes qui m'ont côtoyées tout au long de mes études à l'école et tout particulièrement durant les trois derniers mois de diplôme : Alexandre Gavin, Alexandre Da Costa, Samir Hammoudi et Kenza Majbar avec qui j'ai eu énormément de plaisir à partager mes idées à propos de sujets très variés, Leticia Mikiela qui m'a donné de la motivation et du courage pour faire ce travail et avec qui j'ai pu discuter librement avec infiniment de plaisir, et enfin Julien Probst et Grégoire Wuillamoz avec qui le travail en équipe a été très agréable et très instructif.

Je remercie également chaleureusement tous les étudiants et professeurs que je n'ai pas cités et qui ont fait de mes études un moment très plaisant à passer, en particulier les étudiants de la classe TR 2007 ainsi que les professeurs de l'IICT qui ont donné leur cours avec beaucoup d'entrain et de clarté.

Enfin, un très grand merci à ma famille qui m'a toujours soutenue.

2. *Cahier des charges*

2.1 *Résumé du problème*

Dans le cadre de la plate-forme iminet, un nombre important de fonctions sont implémentées sur le modèle client-serveur, avec un serveur d'applications JBoss et des clients légers (Internet Explorer, Firefox...) abritant les fonctions d'interfaçage avec l'utilisateur.

Pour fonctionnel et flexible que soit ce modèle, il montre ses limites dès que l'on veut implémenter des applications plus gourmandes en interactions avec l'utilisateur. D'autre part, dès que les applications deviennent multi-sessions, l'utilisateur se voit pénalisé par les contraintes de sécurité qui lui imposent de réintroduire son mot de passe pratiquement à chaque nouvelle interaction avec le serveur. Les facilités basées sur les cookies ou sur le stockage en local des mots de passe et implémentées par les navigateurs ou par les systèmes d'exploitation ne sont en principe pas compatibles avec des systèmes à sécurité accrue, comme c'est le cas des systèmes à vocation médicale, et manipulant des données personnelles sensibles.

Un autre inconvénient qui se fait jour est la difficulté qu'il y a de s'adapter à différents éléments d'affichage. Si en théorie, on peut utiliser des transformations XSLT pour adapter la page à un affichage de petites dimensions, il n'en reste pas moins que la notion de formulaire avec laquelle sont implémentées la plupart des entrées de données sur le Web s'accommode assez mal d'écrans de petites dimensions (640 x 320, voire 320 x 240). De plus, la nécessité de rafraîchissement d'une page entière est pénalisante dans le cas de liaisons à bas débit (GPRS), et AJAX, s'il permet de résoudre des problèmes d'interactivité, n'est pas particulièrement économe en bande passante. Or, pour un médecin, l'utilisation d'un ordinateur de poche (PDA, Personal Digital Assistant) comme organe d'entrée-sortie constitue une alternative intéressante, car il peut en principe l'avoir toujours sur lui, et ainsi rester atteignable en urgences sans pour autant être assigné à résidence à proximité d'un ordinateur, ou contraint de promener une lourde valise contenant un ordinateur portable soumis à une séquence de boot-up plus ou moins fastidieuse.

En conséquence, on se propose d'implémenter un client dédié pour PDA, utilisant les protocoles Web (HttpRequest / HttpResponse) mais implémentant son propre moteur d'affichage, dédié à l'application, résident sur le PDA. Il devrait être possible d'ajouter des applications au cours du temps, lesquelles pourront profiter des services offerts par le client dédié, (par exemple le téléchargement de nouvelles applications ou la mise à jour d'applications existantes) mais avec un confort d'utilisation incomparablement plus élevé, et une interface homme-machine (HMI) optimisée pour la plate-forme mobile.

2.2 *Cahiers des charges*

Il s'agit dans ce cadre d'implémenter une infrastructure de sécurisation, de téléchargement, de mise à jour et de lancement d'applications veillant à ce qu'une application donnée ne puisse pas aisément être détournée de son usage premier (cheval de Troie, ou application patchée par décompilation de midlet, par exemple). Cette infrastructure fournira aussi quelques services d'interface avec le PDA (par exemple une interface avec le carnet de rendez-vous ou avec la base de données de contacts) à titre d'exemple et de services aux applications. Enfin, l'infrastructure comprendra un serveur d'applications de support, contrôlant les paramètres utilisés par le terminal mobile, ainsi que les applications qui ont le droit de s'exécuter à un instant donné.

Il en résulte de ce fait une application formée de trois parties bien distinctes :

1. Sécurisation du poste mobile, sécurisation et mise à jour des applications, authentification des utilisateurs
2. Services locaux, applications s'exécutant sur le mobile
3. Services distants, parties « serveur » des applications du bureau mobile

Ces trois parties seront traitées chacune par un étudiant, à savoir :

1. Stephen Badan ETR
2. Grégoire Wuillamoz ETR
3. Julien Probst ETR

3. Table des matières

1.Remerciements.....	3
2.Cahier des charges.....	4
2.1 Résumé du problème.....	4
2.2 Cahiers des charges.....	4
4.Résumé.....	8
5.Introduction.....	9
6.Travail à réaliser dans le cadre du projet de diplôme.....	10
6.1 Tâches réalisées lors du travail de semestre.....	10
6.2 Travail à réaliser lors du projet de diplôme.....	10
7.Use Case.....	12
8.Schémas de principes du service.....	14
8.1 Structure générale.....	14
8.2 Structure détaillée du service dans son ensemble.....	16
8.3 Structure détaillée du service de sécurité.....	18
9.Technologies utilisées.....	20
9.1 Au niveau du client.....	20
9.2 Au niveau du serveur.....	22
10.Service d'authentification – fonctionnement théorique.....	24
10.1 Authentification formelle.....	24
10.2 Authentification matérielle.....	24
10.3 Authentification externe.....	25
10.4 Gestion des méthodes d'authentification.....	26
10.5 Exemples de scénarios d'utilisation.....	30
11.Service d'authentification – réalisation pratique.....	32
11.1 Service de gestion des authentifications.....	32
11.2 Implémentation du service d'authentification formelle.....	35
11.3 Implémentation du service d'authentification matérielle.....	36
11.4 Implémentation du service d'authentification externe.....	40
11.5 Résumé des types d'authentification implémentés.....	41
12.Système de mise à jour.....	42
12.1 Définition d'une application.....	42
12.2 Types de mises à jour.....	43
12.3 Délai de transition.....	43
12.4 Contrôle des versions.....	44
12.5 Types d'installation.....	44
12.6 Contraintes de mise à jour.....	45
12.7 Annulation des téléchargements.....	46
12.8 Nettoyage des fichiers.....	46
12.9 Chargeur d'applications.....	47
12.10 Gestion des mises à jour.....	47
13.Communications entre le client lourd et le serveur.....	50
13.1 Format des messages.....	50
13.2 Sécurisation des communications.....	52
13.3 Identifiant de sécurité.....	54
14.Service d'administration.....	58
14.1 Architecture.....	58
14.2 Sécurité.....	58

14.3 Fonctionnalités.....	59
14.4 Structuration.....	59
14.5 Mise en page.....	60
14.6 Contrôles de saisie.....	60
15.Base de données.....	61
15.1 Modèle conceptuel.....	61
15.2 Modèle relationnel.....	63
16.Diagrammes UML des classes.....	65
16.1 Classes du client lourd.....	65
16.2 Classes du serveur.....	68
17.Rétrospective.....	71
17.1 Retour sur la planification.....	71
17.2 Retour sur le cahier des charges.....	74
17.3 Améliorations possibles.....	74
17.4 Problèmes connus.....	76
18.Conclusion.....	77
19.Annexes.....	78
19.1 Tests.....	78
19.2 Manuel d'utilisation du client lourd.....	92
19.3 Manuel d'utilisation du service d'administration.....	100
19.4 Intégration des différentes parties du client lourd.....	109
19.5 Installation des outils de développement.....	111
19.6 Création des certificats	113
19.7 Installation du service.....	115
19.8 Journal de travail.....	120
20.Lexique.....	124
21.Index des figures.....	125
22.Index lexical.....	127
23.Liste des références.....	129

4. Résumé

"The only truly secure system is one that is powered off, cast in a block of concrete and sealed in a lead-lined room with armed guards – and even then I have my doubts."¹

Eugene H. Spafford [1]

Cette citation, très connue dans le domaine des nouvelles technologies, illustre à quel point la sécurité informatique et des données en général est difficile à assurer. Cette problématique est relativement récente, il est en effet possible depuis quelques années de centraliser une quantité impressionnante d'informations plus ou moins confidentielles dans un seul ordinateur, voire même aujourd'hui dans un simple téléphone mobile.

Qu'en est-il alors de la sécurité ? Ce projet se propose d'étudier ladite question en implémentant une application permettant d'assurer au mieux la sécurité de données contenues dans un appareil mobile. Cette application, nommée client lourd, sera installée sur téléphone mobile ou un assistant personnel électronique et devra gérer la sécurité d'accès et la mise à jour d'applications médicales installées sur ce client. L'objectif de ces applications médicales, développées dans le cadre de deux autres travaux de diplôme, est d'offrir aux médecins des outils de consultation de fiches de patients à distance, ainsi que des outils de gestion d'alertes provenant de personnes victimes de problèmes médicaux, tels que des malaises. Ce client lourd permettra ainsi aux médecins de travailler en déplacement tout en gardant un haut niveau de sécurité.

Certaines personnes se demanderont alors à juste titre de quelle manière la sécurité peut être assurée ? La première partie de ce dossier répondra à cette question en montrant comment un système d'authentification forte peut empêcher les accès indésirables à un appareil mobile. Ce système sera composé d'une authentification matérielle de l'appareil mobile, d'une authentification formelle de l'utilisateur et d'une authentification externe utilisant la technologie Bluetooth.

Mais la sécurité ne s'arrête pas là : les applications qui seront automatiquement installées sur le client lourd pourront être exposées au risque de ne plus être à jour lors de la publication de nouvelles versions sur le serveur. Il s'ajoute à cela que ces applications pourraient être copiées, puis modifiées par des personnes malintentionnées dans le but de nuire à la confidentialité ou à l'intégrité des données. La seconde partie de ce dossier se concentrera ainsi sur le service de mise à jour qui assurera que les applications installées soient valides et à jour afin d'éviter tout problème de ce type.

Ce dossier comprendra également les explications du fonctionnement en lui-même de ce client lourd et de son serveur de sécurité, car les paramétrages possibles sont nombreux. Divers schémas, manuels d'explications et guides d'installation permettront par ailleurs de faciliter la compréhension d'un tel service de sécurité.

Enfin, ce projet se terminera par une rétrospective du déroulement général de ce travail de diplôme, des fonctionnalités qu'il serait possible d'ajouter ou d'améliorer et des éléments essentiels qu'il a été possible d'en retenir.

Au vu de ce qui précède, il ne me reste maintenant qu'à vous souhaiter une très agréable lecture.

¹ « Le seul système vraiment sécurisé est un système éteint, coulé dans un bloc de béton et scellé dans une chambre doublée de plomb surveillée par des gardes armés – et alors même j'ai mes doutes. »

5. Introduction

Durant la dernière décennie, la science a permis de rendre mobile des technologies qui étaient jusqu'alors limitées à des appareils lourds et encombrants : que ce soit la télévision, la visiophonie, la musique, l'accès aux cours de la bourse ou à Internet de manière générale, toutes ces fonctionnalités sont maintenant à la portée d'un simple téléphone mobile, et ce, où que l'on soit. Cette avancée technologique ouvre de nouvelles portes dans le monde de l'information où nous ne sommes dorénavant plus limités à un emplacement physique pour effectuer une tâche donnée, pour peu que nous possédions une connexion à un réseau sans fil.

En outre, l'essor des appareils mobiles, tels que les téléphones ou assistants personnels, a permis de les doter d'une puissance de calcul et d'une mémoire suffisantes pour exécuter des applications complexes qui étaient habituellement réservées à des ordinateurs fixes. C'est dans ce cadre que l'on se propose de développer un client lourd spécialisé dans le domaine de la santé, mais aussi utilisable dans d'autres domaines, tels que le domaine bancaire ou le domaine financier.

Bien que la notion de client lourd puisse au premier abord paraître assez absconse, elle pourrait être comparée à une plateforme sur laquelle pourront venir s'ajouter d'autres applications ayant des fonctionnalités très spécifiques, telles que la gestion de clients ou de rendez-vous. Le client lourd s'occuperait alors de gérer l'exécution, la sécurité, les mises à jour et les communications de ces applications.

Il vient alors la question de l'utilité d'un tel client lourd ? Pour y répondre, imaginons que des applications soient spécialement développées pour le milieu médical. Il serait alors possible pour un médecin de pouvoir accéder, au travers des applications du client lourd, à toutes les informations médicales concernant ses patients, voire même recevoir des alertes lorsque certains de ceux-ci sont victimes d'un malaise. En rendant ce client lourd mobile, on offre au médecin la possibilité d'effectuer toutes ces tâches sans limites géographiques, rendant le service extrêmement flexible.

Cette souplesse engendre en revanche des questions légitimes concernant la sécurité des données. Que se passerait-il si l'appareil mobile du médecin était volé ? Quelqu'un de malintentionné pourrait-il accéder aux données confidentielles ? Les attaques possibles sont en effet multiples et les conséquences peuvent être critiques.

Le but de ce travail de diplôme est de répondre à ces questions en réalisant un service de sécurité pour le client lourd, ayant pour but de protéger les applications et l'accès aux informations afin de contrer au mieux les différents risques qu'il est envisageable de rencontrer.

6. Travail à réaliser dans le cadre du projet de diplôme

6.1 Tâches réalisées lors du travail de semestre

Lors du travail de semestre, nommé aussi pré-projet de diplôme, le but a été de s'informer sur les technologies à disposition et d'étudier la faisabilité du cahier des charges dans le but de le redéfinir si nécessaire. Une fois cette étude terminée, il s'agissait de concevoir les schémas du service et de planifier les tâches à effectuer dans le cadre du travail de diplôme. Aucune implémentation des fonctionnalités n'a cependant été effectuée durant le travail de semestre, à moins que celle-ci n'ait été réalisée qu'à des fins de tests de faisabilité.

Le pré-projet de diplôme a ainsi permis de définir précisément les tâches à effectuer dans le cadre du travail de diplôme.

6.2 Travail à réaliser lors du projet de diplôme

Le projet de diplôme comprend la réalisation des tâches définies lors du travail de semestre. Deux types de tâches ont été définis : les tâches prioritaires qui doivent absolument être terminées d'ici à la fin du projet et les tâches optionnelles qui peuvent apporter de nouvelles fonctionnalités, mais qui ne sont réalisées que si le temps le permet.

6.2.1 Tâches prioritaires à effectuer

Cette liste de tâches définit de manière très précise les différentes tâches à effectuer obligatoirement.

- ◇ Tâches à effectuer au niveau du client lourd :
 - Pour tous les services :
 - Implémenter l'envoi, le traitement et la réception de documents XML à l'aide d'un parseur XML;
 - Service de mise à jour :
 - Implémenter la réception de fichiers;
 - Implémenter le stockage et le chargement des applications sur le client lourd;
 - Implémenter le contrôle des applications et la gestion des mises à jour de celles-ci au démarrage du client lourd;
 - Service de sécurité :
 - Mettre en place une communication sécurisée entre le client mobile et le serveur de sécurité à l'aide du protocole SSL;
 - Implémenter le service d'authentification formelle (par mot de passe);
 - Implémenter le service d'authentification matérielle;
 - Implémenter la gestion des demandes d'authentification (tableau de la page 29);
 - Implémenter le contrôle des applications (temporisateurs et droits d'accès);
- ◇ Tâches à effectuer au niveau du serveur de sécurité
 - Pour tous les services :

- Mettre en place la base de données et la classe d'accès à celle-ci;
- Implémenter l'envoi, le traitement et la réception de documents XML à l'aide de JAXB;
- Service de mise à jour :
 - Implémenter le contrôle des nouvelles mises à jour en tenant compte des droits de l'utilisateur;
 - Implémenter l'envoi de fichiers au client lourd;
- Service de sécurité :
 - Implémenter l'attribution d'identifiants temporaires de sécurité;
 - Implémenter le contrôle des authentifications et des droits de l'utilisateur;
- Service d'administration :
 - Implémenter les ajouts, modifications et suppressions d'utilisateurs, programmes, appareils mobiles ou paramètres de sécurité;
- ◇ Tâches à effectuer au niveau de l'interface Web d'administration :
 - Mettre en place une communication sécurisée entre le navigateur Internet de l'administrateur et le serveur de sécurité à l'aide du protocole SSL;
 - Implémenter une page d'authentification formelle de l'administrateur;
 - Implémenter les outils d'ajouts, modifications ou suppressions d'utilisateurs, appareils mobiles, groupes, paramètres de sécurité et programmes;
 - Implémenter l'envoi de nouvelles applications au serveur par l'administrateur;

6.2.2 Tâches optionnelles

Ces différentes tâches seront réalisées en fin de projet en fonction du temps restant. Elles peuvent par ailleurs être modifiées, voire supprimées, en fonction de l'évolution du projet lors de sa réalisation.

Cette disposition des tâches permet de se concentrer en premier lieu aux fonctionnalités prioritaires, puis aux fonctionnalités utiles, mais sans lesquelles le service est utilisable. Ces tâches ne sont pas représentées dans le micro-calendrier, mais s'effectueraient dans la semaine précédant la finalisation du rapport.

- ◇ Tâches à effectuer au niveau du client lourd en collaboration avec le serveur de sécurité
 - Service de mise à jour :
 - Implémenter la possibilité d'interruption d'un transfert d'une application;
 - Permettre de mettre à jour des applications pendant le fonctionnement du client lourd;
 - Service de sécurité
 - Implémenter une simulation d'authentification externe de type biométrique avec l'algorithme de contrôle de GrFinder (voir GrFinder Java 4.5, chapitre 11.4, page 40);

7. Use Case

Afin de définir clairement les fonctionnalités qui devront être implémentées dans ce projet, celles-ci sont décrites dans cette partie, tout en précisant en quoi consiste chaque fonctionnalité et quel acteur est autorisé à l'exécuter.

Dans cette partie du projet qui se concentre essentiellement sur la sécurité, deux acteurs principaux seront concernés :

- ◇ Le médecin qui utilise l'appareil mobile;
- ◇ L'administrateur du service qui gère le service de sécurité;

Les diverses fonctionnalités à implémenter dans ce projet sont illustrées dans le diagramme de contexte des cas d'utilisation ci-dessous :

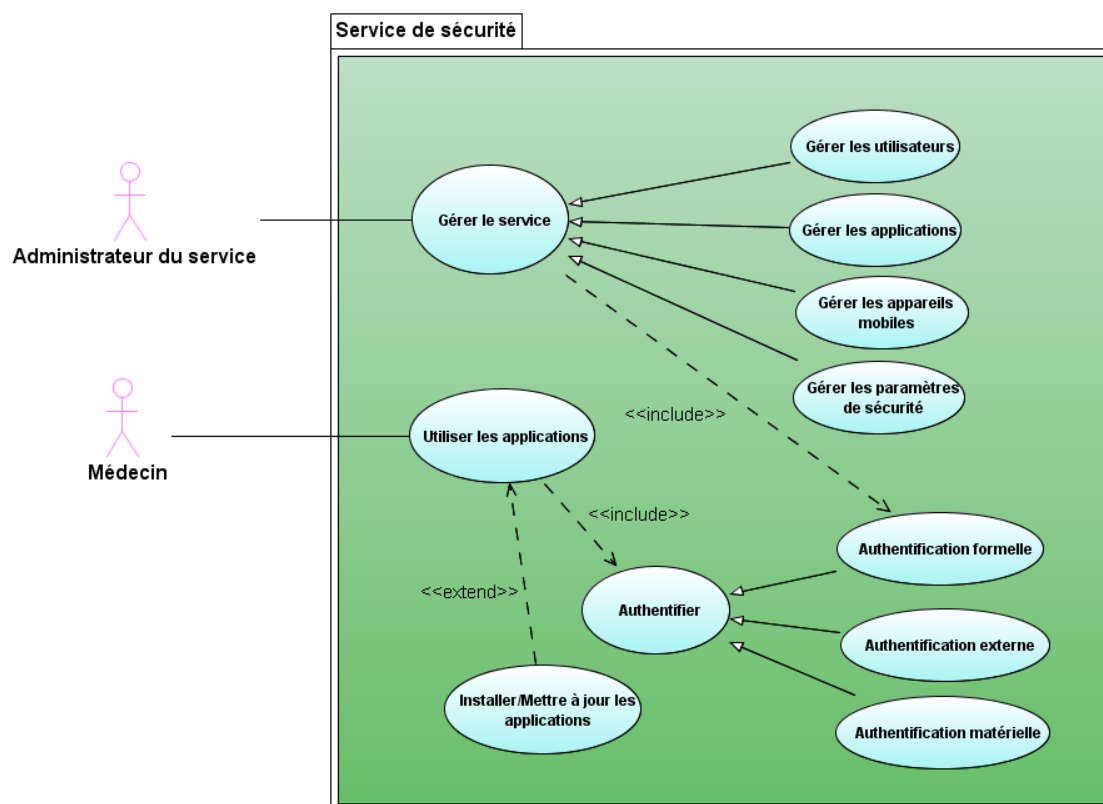


Figure 1: Diagramme de contexte du service de sécurité

Description des fonctionnalités destinées au médecin :

- ◇ Utiliser les applications : comme son nom l'indique, cette fonctionnalité n'est autre que l'utilisation des applications qui seront mises à disposition du médecin dans le client lourd. Pour préserver l'intégrité du service et la protection des données, l'utilisation des applications nécessitera que le médecin s'identifie et s'authentifie avant de pouvoir les utiliser.
- ◇ Mettre à jour des applications : cette fonctionnalité permettra de garder les applications du client lourd à jour en fonction des applications stockées sur le serveur et d'assurer leur intégrité.
- ◇ Authentifier : cette fonctionnalité permettra d'identifier et d'authentifier les médecins afin de leur

donner ou non l'accès au service désiré. Cette étape est obligatoire pour toute utilisation du service. Cette fonctionnalité est décomposée en trois sous-fonctionnalités :

- Authentification formelle : cette authentification se base sur un mot de passe textuel pour authentifier l'utilisateur.
- Authentification matérielle : cette authentification se base sur des caractéristiques matérielles permettant d'authentifier l'appareil utilisé, par exemple un identificateur unique.
- Authentification externe : cette authentification se base sur un élément externe qui permet d'authentifier le médecin. Cette authentification externe peut être une authentification géographique du médecin et de son appareil mobile ou encore une authentification biométrique du médecin.

Ces différents types d'authentifications sont décrits de manière détaillée dans le chapitre 10 «Service d'authentification – fonctionnement théorique» page 24 et dans le chapitre 11 «Service d'authentification – réalisation pratique» page 32.

Description des fonctionnalités destinées à l'administrateur du service :

- ◇ Gérer les utilisateurs : l'administrateur du service pourra ajouter, modifier ou supprimer des utilisateurs ainsi que paramétrer leur mot de passe ou autre clé d'authentification externe.
- ◇ Gérer les applications : l'administrateur pourra ajouter, modifier ou supprimer des applications. Il pourra également attribuer les droits d'utilisation des applications aux utilisateurs.
- ◇ Gérer les appareils mobiles : tous les appareils mobiles utilisés dans le service pourront être gérés par l'administrateur. Il pourra définir quels utilisateurs ont le droit d'utiliser un appareil donné.
- ◇ Gérer les paramètres de sécurité : comme son nom l'indique, cette tâche consiste à définir et à gérer les paramètres des authentifications ou des mises à jour.

La gestion du service requiert également une authentification de l'administrateur avant de pouvoir être utilisée.

8. Schémas de principes du service

Le service de sécurité sera composé de plusieurs sous-services bien distincts. Comme plusieurs personnes travaillent sur le même projet, il faut porter une importance particulière au découpage des parties afin qu'elles soient aussi indépendantes que possible.

Les trois schémas qui suivent se développeront du plus général au plus détaillé.

8.1 Structure générale

Dans un premier temps, il est intéressant de considérer le service dans son ensemble. La figure 2 «Schéma général du service», permet d'avoir un premier aperçu de la structure générale du service.

Le service sera divisé en trois parties principales :

- ◇ La première partie représente le client lourd lui-même. Ce client contient le gestionnaire de sécurité et les applications installées.
- ◇ La seconde partie est le serveur de sécurité qui permet de gérer les demandes d'authentification des clients mobiles, ainsi que la gestion des envois des applications vers les clients lourds.
- ◇ La troisième partie correspond au(x) serveur(s) d'applications. Chaque application a la possibilité d'avoir un serveur dédié, ce qui permet de placer physiquement ce type de serveurs dans un endroit différent de l'emplacement du serveur de sécurité. De ce fait, il est possible d'avoir plusieurs serveurs d'applications, chaque serveur pouvant gérer une ou plusieurs applications.

Les communications entre le client mobile et les serveurs se font de manière sécurisée, à l'aide d'un chiffrement des données par TLS (*Transport Layer Security*), anciennement nommé SSL (*Secure Sockets Layers*).

De manière plus détaillée, le client mobile possède trois composants distincts :

- ◇ Le gestionnaire de sécurité qui gère les demandes d'authentification ainsi que les mises à jour des applications;
- ◇ Les applications qui offrent les fonctionnalités du client lourd pour l'utilisateur;
- ◇ Les modules de sécurité qui font office de liens entre le gestionnaire de sécurité et les applications exécutées. Un contrôle doit s'effectuer lorsque des requêtes sont envoyées par les applications. Ce contrôle vérifie si l'utilisateur a le droit d'utiliser l'application et si une authentification est nécessaire afin de pouvoir continuer à l'utiliser.

Mon travail de diplôme concerne l'implémentation du serveur de sécurité ainsi que l'implémentation du gestionnaire de sécurité au niveau du client. Le serveur d'applications et l'implémentation des applications elles-mêmes font l'objet de deux autres travaux de diplôme.

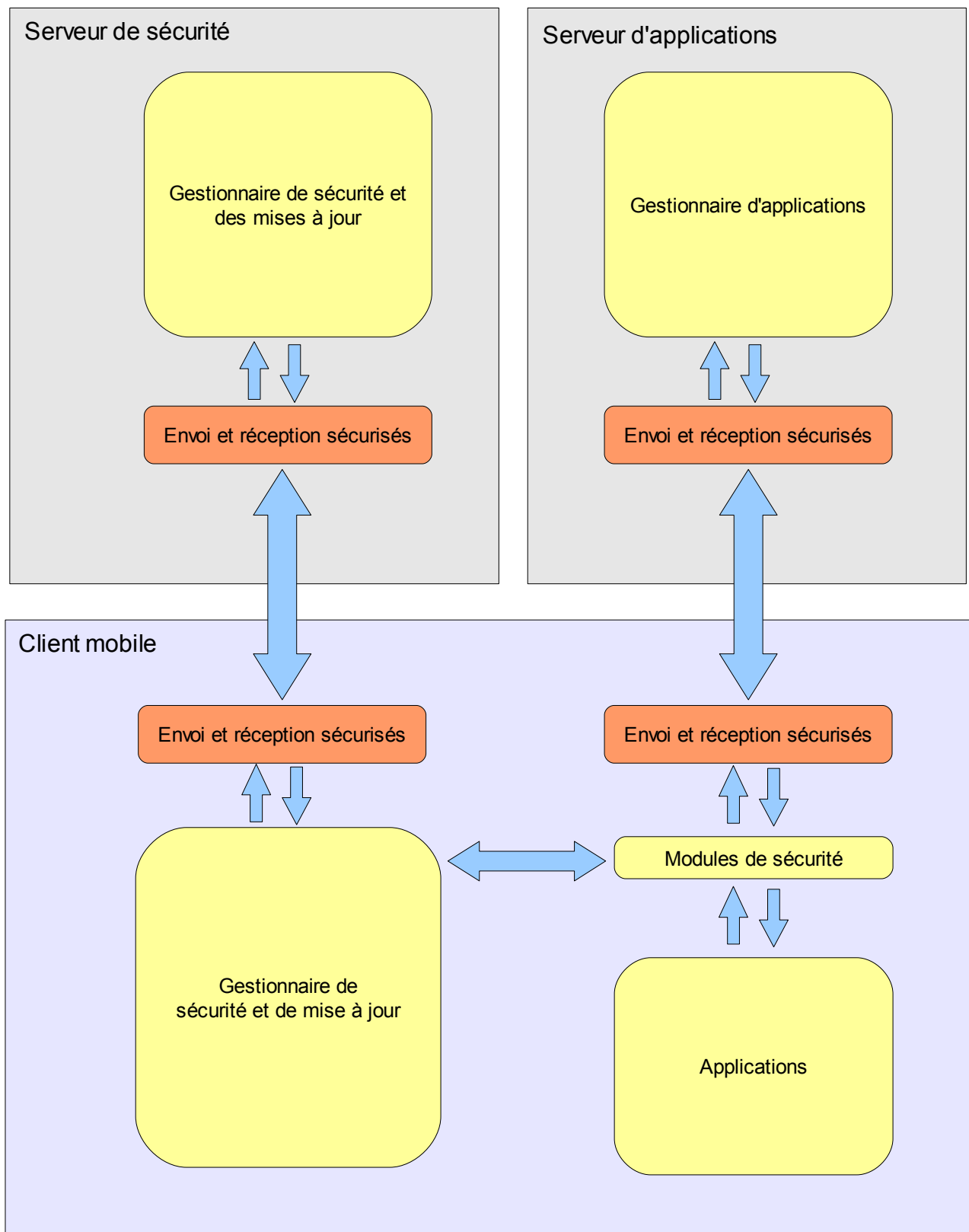


Figure 2: Schéma général du service

8.2 Structure détaillée du service dans son ensemble

La figure 3 «Structure détaillée du service» permet de détailler chacune des différentes parties du service.

Le serveur de sécurité est séparé, en interne, en trois parties distinctes : le service des mises à jour, le service de sécurité et le service d'administration :

- ◇ Le gestionnaire de sécurité gère toutes les demandes d'authentification provenant des clients ainsi que les contrôles des droits des utilisateurs et des paramètres de sécurité.
- ◇ Le gestionnaire des mises à jour s'occupe de transférer les applications vers les clients et collabore avec le gestionnaire de sécurité pour vérifier les droits d'utilisation des applications par les utilisateurs lors des mises à jour.
- ◇ Le service d'administration permet à l'administrateur de définir et de gérer les différents paramètres de sécurité, tels que les comptes d'utilisateurs, les comptes des appareils mobiles, les applications installées, les droits et les paramètres des authentifications.

Au niveau du client lourd, le gestionnaire de sécurité et des mises à jour a été scindé en trois parties distinctes :

- ◇ Le gestionnaire des mises à jour s'occupe de vérifier l'état et la version des applications installées et effectue des mises à jour et des installations d'applications si nécessaire. Il contrôle aussi l'intégrité des applications installées.
- ◇ Le gestionnaire des demandes d'authentification effectue celles-ci en fonction des paramètres de sécurité et vérifie que les applications exécutées par l'utilisateur ont le droit d'être utilisées par ce dernier.
- ◇ Le service des authentifications contient les différentes méthodes d'authentification qu'il est possible d'utiliser sur le client mobile. Ce service est appelé par le gestionnaire des demandes d'authentification lorsqu'une authentification est nécessaire, tout en précisant quelle méthode d'authentification est souhaitée. Une fois appelé, ce service se charge d'exécuter l'authentification demandée et de retourner la réponse au gestionnaire des demandes d'authentification.

Au niveau des applications sur le client mobile, chaque application possède un gestionnaire des requêtes qui gère les communications entrantes et sortantes. Ce gestionnaire communique également avec un module de sécurité qui fait office de lien entre l'application et le gestionnaire des demandes d'authentification. Ce lien permet d'inclure un identifiant de sécurité dans la requête lorsqu'une application souhaite envoyer un message à son serveur dans le but de sécuriser le service². Enfin, le chargeur d'applications permet de charger et d'exécuter les applications stockées sur le client lourd.

² Toutes les informations concernant l'identifiant de sécurité sont données dans le chapitre 13.3, page 54.

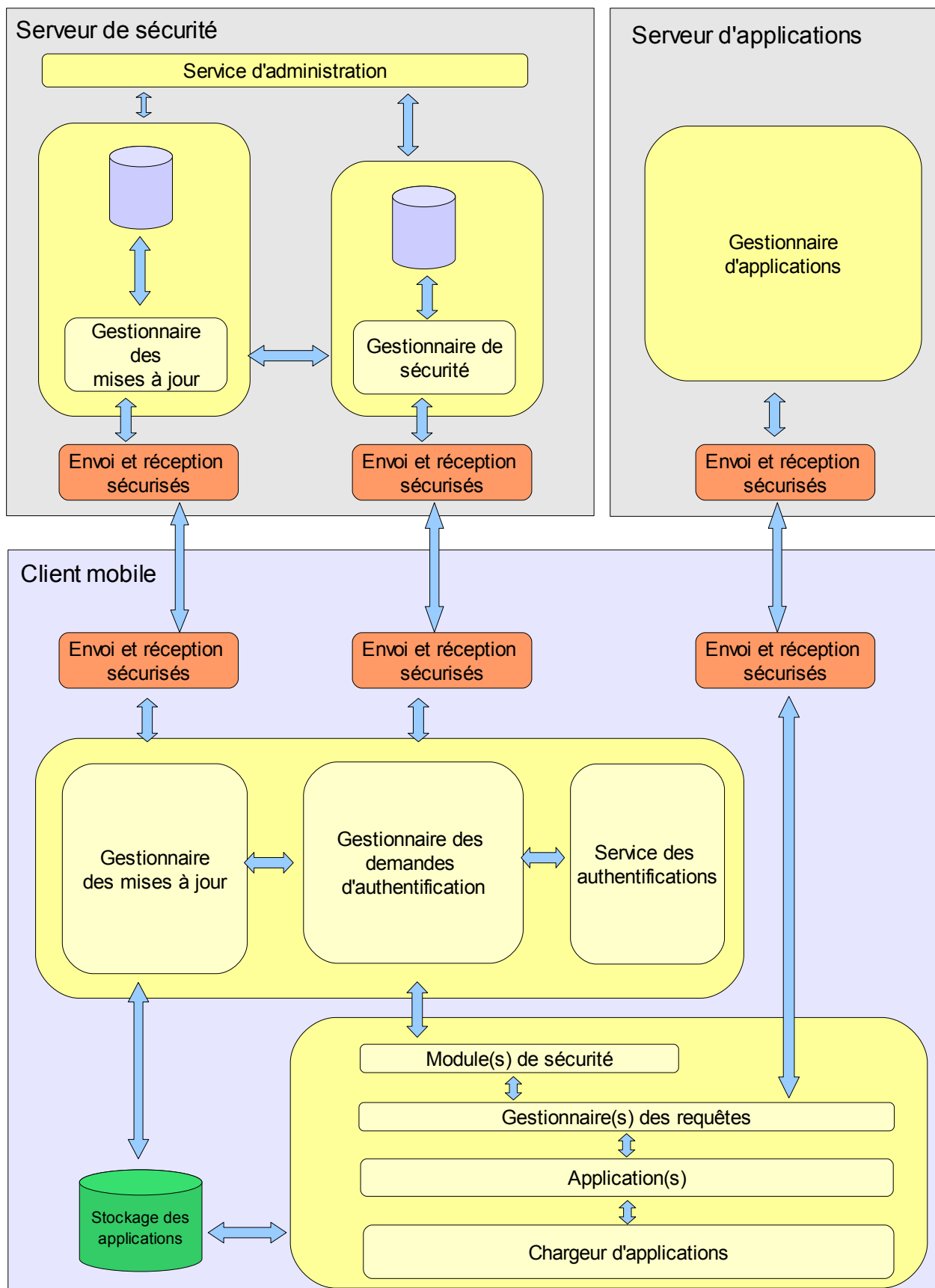


Figure 3: Structure détaillée du service

8.3 Structure détaillée du service de sécurité

La figure 4 «Structure détaillée du service de sécurité» ne tient plus compte des applications et de leur serveur qui sont traités dans le cadre d'autres travaux de diplôme. Le schéma ne détaille que le service de sécurité, tant au niveau du client qu'au niveau du serveur.

Au niveau du serveur de sécurité, nous retrouvons le gestionnaire des mises à jour et le gestionnaire de sécurité décrits précédemment. Concernant le stockage des informations, deux nouvelles entités ont été détaillées :

- ◇ Le serveur de fichiers : ce serveur contient tous les fichiers des applications qui peuvent être transférées sur les clients;
- ◇ Le système de gestion de base de données (SGBD) : ce serveur contient toutes les données concernant les utilisateurs, les appareils mobiles, les droits et les paramètres de sécurité. Ce serveur contient également les informations concernant les applications, telles que la date d'ajout, les droits d'utilisation des applications par les groupes d'utilisateurs, etc.

Ces deux serveurs (« Modèles ») possèdent chacun une classe d'accès qui permet aux gestionnaires de sécurité et de mise à jour (« Contrôleurs ») d'accéder aux informations. Ce découpage a été fait afin de respecter l'architecture MVC (Modèle – Vue – Contrôleur) qui permet d'assurer une grande indépendance entre les différentes couches dans le but de pouvoir les modifier ou les remplacer aisément, sans avoir à modifier les autres couches.

Le service d'administration (« Contrôleur ») permettra à l'administrateur de se connecter par Internet au serveur de sécurité afin de gérer tous les paramètres de sécurité du service et de gérer les applications. Pour ce faire, une interface Web (« Vue ») est implémentée.

Du côté du client mobile, nous retrouvons les deux parties principales (« Contrôleurs ») qui sont la partie de mise à jour et la partie de gestion des authentifications. Le service de gestion des authentifications a par ailleurs été décomposé en deux parties :

- ◇ Le gestionnaire des demandes d'authentification qui effectuent les authentifications en fonction des paramètres de sécurité;
- ◇ Le contrôle des applications qui contient la liste des applications en cours d'exécution et qui effectue les vérifications des droits d'utilisation des applications par l'utilisateur lorsque celui-ci démarre une nouvelle application;

Le service des authentifications a aussi été scindé en trois services d'authentification distincts :

- ◇ Le service d'authentification formelle implémente toutes les méthodes nécessaires pour authentifier l'utilisateur à l'aide d'un nom et d'un mot de passe.
- ◇ Le service d'authentification externe authentifie l'utilisateur à l'aide d'un élément externe, telle qu'une authentification biométrique ou une authentification géographique de l'utilisateur et de son appareil mobile.
- ◇ Le service d'authentification matérielle authentifie l'appareil mobile utilisé à l'aide d'un code d'identification unique.

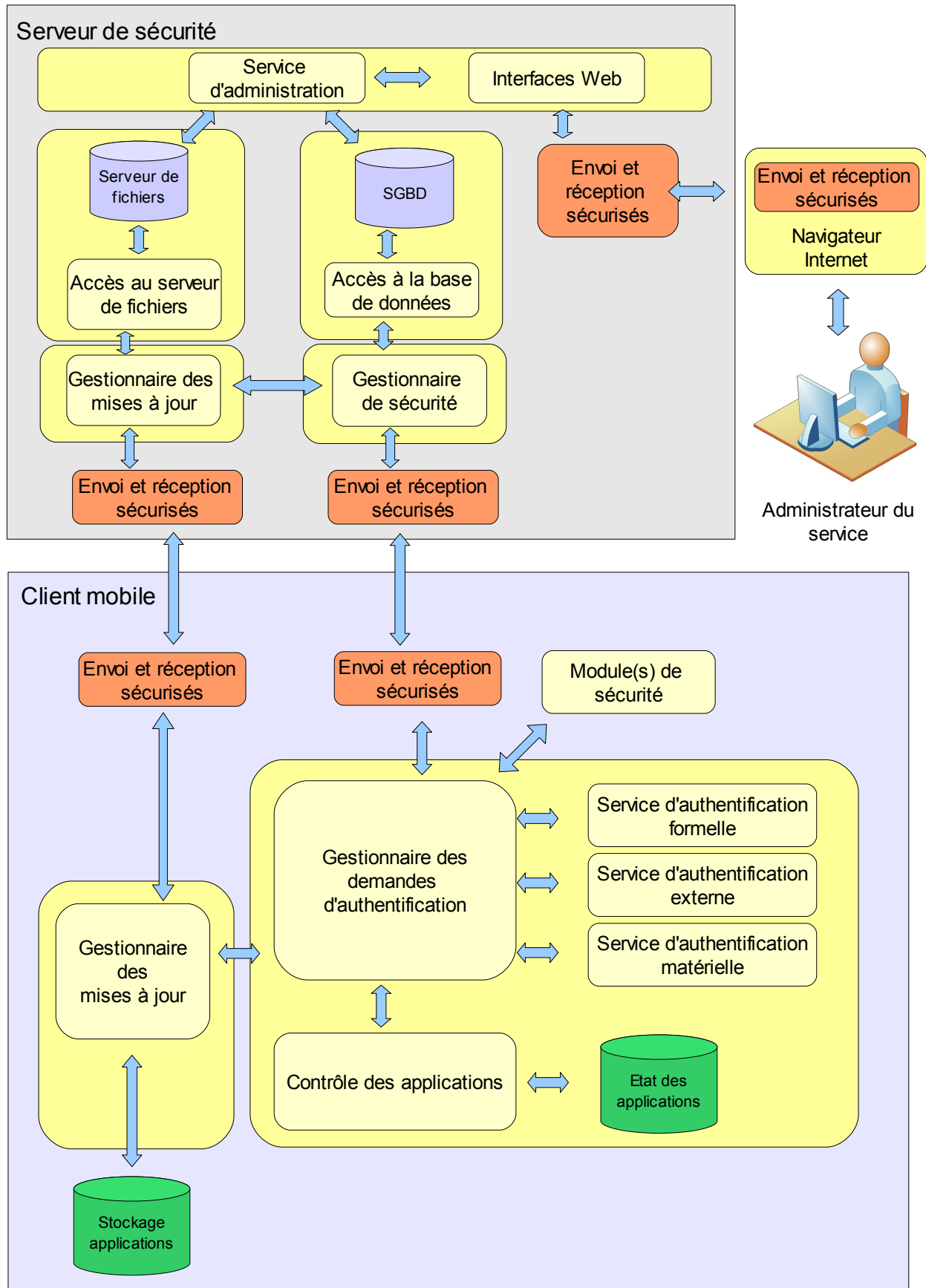


Figure 4: Structure détaillée du service de sécurité

9. Technologies utilisées

9.1 Au niveau du client

9.1.1 Langage de programmation

Le client lourd est écrit dans le langage Java. Comme ce client doit s'exécuter dans un environnement mobile comme des PDA (*Personal Digital Assistant*), il n'est pas possible d'utiliser la version standard de Java (Java SE – *Java Standard Edition*) en raison des besoins trop élevés en ressources systèmes. Sun propose une version réduite de Java nommée Java ME (*Java Micro Edition*) qui peut s'exécuter sur des environnements à ressources limitées. Java ME contient deux sous-catégories principales qui sont CDC (*Connected Device Profile*) et CLDC (*Connected Limited Device Configuration*). [2] [3]

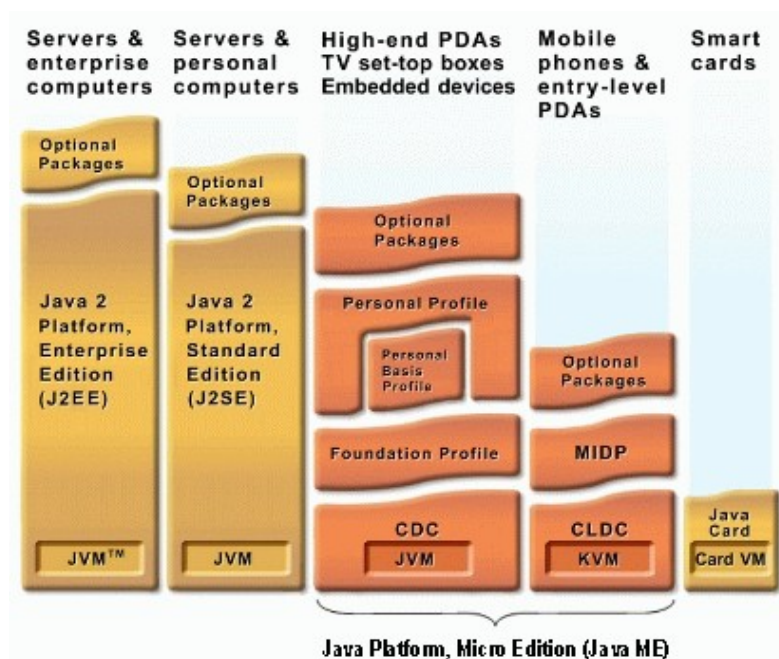


Figure 5: Profiles Java

Les configurations CDC et CLDC possèdent trois éléments principaux :

- ◇ Une configuration qui contient les fonctionnalités les plus basiques pour un grand nombre d'appareils mobiles;
- ◇ Un profil contenant des API (*Application Programming Interface*), ou fonctionnalités, plus spécifiques aux types d'appareils utilisés;
- ◇ Un paquetage optionnel contenant des API très spécifiques à certaines technologies;

Le profil CLDC a été créé pour fonctionner sur des appareils mobiles à ressources très limitées (moins de 512 Ko de mémoire), tels que des téléphones portables. La couche MIDP (*Mobile Information Device Profile*) offre un environnement Java complet pour ce type d'appareils et ce type de contraintes.

Quant au profil CDC, celui-ci a été prévu pour fonctionner sur des appareils mobiles ayant des ressources moins limitées (plus de 2 Mo de mémoire), telles que des PDA ou des téléphones mobiles de type *Smartphones* (téléphones « intelligents »). Le profil CDC contient quant à lui trois couches distinctes :

- ◇ Un profil de base nommé *Foundation profile* qui contient des API fonctionnant sans interface graphique;
- ◇ Un profil nommé *Personal Basis Profile* contenant des API fonctionnant avec des interfaces graphiques à composants légers (tels que Swing par exemple);
- ◇ Un profil nommé *Personal Profile* qui est une extension de *Personal Basis Profile* offrant des fonctionnalités pour des interfaces graphiques à composants lourds (AWT, SWT) et pour les applets;

Au final, le but de Java ME CDC est d'offrir un environnement Java proche de Java SE, tout en fonctionnant sur des appareils à ressources relativement limitées.

Dans le cadre de ce projet, j'ai choisi la configuration Java ME CDC, car celle-ci se rapproche le plus de la version standard de Java et offre ainsi un plus grand nombre de bibliothèques. Les appareils mobiles utilisés dans ce projet offrent des caractéristiques matérielles suffisantes pour la configuration CDC.

Au niveau des alternatives à Java pour développer les applications pour appareils mobiles, il existe la solution .NET Compact Framework éditée par Microsoft qui permet de créer des applications fonctionnant sur le système d'exploitation Windows Mobile. L'avantage de la solution de Microsoft est qu'elle permet de créer des applications qui utilisent pleinement toutes les fonctionnalités du système d'exploitation. Il est aussi possible d'accéder aux ressources du système par Java ME, mais il faut pour cela utiliser des solutions telles que JNI³ (*Java Native Interface*) compliquant considérablement la tâche. Java ME possède quant à lui tous les avantages de Java, dont celui de fonctionner sur de multiples plateformes telles que Windows Mobile ou Symbian OS, pour autant qu'une machine virtuelle ait été implémentée pour le système d'exploitation utilisé, ce qui n'est pas le cas de .NET qui est limité à la plateforme Windows. Java ME permet par ailleurs de charger du code dynamiquement, ce qui présente un très grand avantage dans le contexte d'un client lourd où le chargement d'applications téléchargées est l'une des principales contraintes. Pour ces différentes raisons, la réalisation de ce projet est faite dans le langage Java ME.

Enfin, à titre d'information, Java ME CDC 1.1 est l'équivalent d'une version 1.4.2 de Java SE épurée, c'est-à-dire sans méthode obsolète (ou *deprecated* dans le contexte Java). [4]

9.1.2 Machine virtuelle

Sun ne propose pas de machine virtuelle pour Java ME. Il faut donc utiliser des machines virtuelles éditées par des entreprises externes. Il n'existe qu'un nombre restreint de machines virtuelles pour Windows Mobile. Parmi les plus connues se trouvent la machine J9 d'IBM incluse dans l'outil de développement *WebSphere Everyplace Micro Environment* d'IBM et la machine CrEme de NSICOM. Ces deux machines virtuelles sont réputées pour être stables, mais sont payantes. Dans le cadre de ce projet, j'ai opté pour la machine virtuelle d'IBM, car celle-ci supporte le profil CDC 1.1 alors que CrEme ne supporte que le profil CDC 1.0. La machine virtuelle d'IBM possède par ailleurs un meilleur support et une meilleure aide que la machine virtuelle CrEme. Une version de démonstration est utilisée dans ce projet. Aucune information officielle n'a malheureusement pu être trouvée quant au coût de la version définitive, le site d'IBM ne donnant aucun prix pour ce produit. Il semblerait toutefois que cette machine virtuelle soit en vente à un prix de 25 US\$ par appareil mobile. [5]

9.1.3 Interface graphique

La machine virtuelle d'IBM inclut par défaut les composants graphiques AWT. Ces composants étant relativement limités (ils ne contiennent que les boutons, champs de saisie et labels), j'ai choisi d'utiliser

3 JNI permet d'exécuter, au sein d'un programme Java, du code C ou C++ contenu dans une bibliothèque externe de type DLL (*Dynamic Link Library*). Un exemple pratique d'utilisation de JNI est donné au chapitre 11.3, page 36.

des composants graphiques externes de type SWT (*Standard Widget Toolkit*).

SWT a été développé par IBM, puis a été repris par l'*Eclipse Foundation* et est sous licence open source. Cette solution est écrite en Java et utilise des fonctionnalités natives à l'aide de JNI afin d'afficher les composants graphiques lourds au niveau du système d'exploitation. L'utilisation de JNI induit l'utilisation d'une librairie externe spécifique à la plateforme utilisée, ce qui explique que chaque version de SWT est spécifique à un seul système d'exploitation. Il existe une multitude de versions de SWT destinées à des plateformes de type différent, dont une version pour Windows Mobile 5.0.

L'alternative à SWT est Swing. Swing est écrit en Java uniquement (pas d'appels JNI, donc composé de composants graphiques légers), ce qui lui offre une meilleure portabilité et une apparence visuelle plus indépendante de la plateforme utilisée et permet aussi de créer des interfaces graphiques avec des hiérarchies de fenêtres plus complexes qu'avec SWT. En revanche, SWT est mieux optimisé pour les appareils à ressources limitées, car l'utilisation de JNI permet d'utiliser les ressources graphiques du système d'exploitation, contrairement à Swing qui fonctionne dans les couches supérieures de Java et qui, après essais, se trouve être trop lourd à l'exécution sur des appareils mobiles. [6] [7]

9.1.4 Échange de messages

Les échanges des messages entre les clients et le serveur s'effectuent à l'aide de fichiers XML. La technologie XML (*eXtensible Markup Language*) est basée sur le langage SGML (*Standard Generalized Markup Language*) et permet de représenter des données à l'aide de balises textuelles. La structure de ces balises est définie par l'utilisateur par l'intermédiaire d'un schéma XSD appelé *XML Schema*. [8]

L'utilisation de fichiers XML permet de structurer les informations à envoyer afin que celles-ci puissent être extraites facilement au moment de la réception des fichiers. La création et la lecture de ces fichiers sont possibles à l'aide d'un parseur XML.

- ◇ Au niveau du serveur, le parseur XML utilisé est JAXB (*Java Architecture for XML Binding*) qui permet de traduire automatiquement les documents XML en objets Java, et vice-versa.
- ◇ Au niveau du client, le parseur manuel kXML2 est utilisé, car JAXB n'est pas disponible pour Java ME.

Les détails concernant le fonctionnement de ces parseurs, la structure du schéma XML et des exemples de fichiers sont donnés dans le chapitre 13.1, page 50.

9.2 Au niveau du serveur

9.2.1 Servlets et JSP

Le serveur utilise le langage Java EE 5 (Java Enterprise Edition version 5) de Sun Microsystems permettant de créer des servlets et des JSP. La technologie Java au niveau du serveur garantit une bonne gestion de la montée en charge du serveur et permet une bonne structuration du service ainsi qu'une faible tolérance aux erreurs, ce qui est particulièrement important dans un service de sécurité.

Une servlet est écrite en Java et permet de générer et d'afficher dynamiquement des données sur un serveur HTTP, comme un site Internet par exemple. Dans le cadre de ce projet, les servlets sont utilisées pour recevoir les requêtes provenant des clients mobiles et pour renvoyer le résultat de leur traitement. Les JSP (*Java Server Pages*) sont quant à elles utilisées lorsque la quantité de code HTML au sein de celles-ci est plus importante que la quantité de code Java, pour des raisons de facilité d'écriture du code. En dehors du confort d'utilisation, les JSP ne diffèrent en rien aux servlets, car les JSP, une fois compilées par le serveur Web, sont totalement équivalentes à des servlets. Dans ce projet, j'utilise les JSP pour tout le service d'administration où le code HTML est majoritairement présent.

9.2.2 Base de données

La base de données utilisée au niveau du serveur est de type MySQL.

Les avantages de MySQL sont multiples [9]:

- ◇ Popularité : MySQL est extrêmement répandu en raison de sa simplicité et de sa flexibilité;
- ◇ Coûts : le serveur MySQL de base est gratuit (*MySQL Community Server*). D'autres versions plus évoluées existent aussi, offrant entre autres un support d'aide pour le client. La version de base est cependant suffisante dans le cadre de ce projet;
- ◇ Performances et fiabilité : ce système de gestion de base de données offre une grande fiabilité et de très bonnes performances, car il bénéficie d'un support de longue date.

Au niveau des inconvénients, MySQL n'offre pas toutes les possibilités fonctionnelles des systèmes de gestion de bases de données professionnelles, telles les performances au niveau des indexations qu'offre Oracle. Les fonctionnalités offertes par MySQL Community Server sont toutefois suffisantes, c'est pourquoi j'ai opté pour cette solution.

9.2.3 Serveur de fichiers

Le serveur de fichiers utilisé pour stocker les applications au niveau du serveur de sécurité est la solution WebDAV. Jakarta Slide permet ensuite d'accéder à un tel serveur.

Jakarta Slide est un système de gestion de contenu utilisant le protocole WebDAV. Slide est écrit en Java et offre un ensemble d'API implémentant un client WebDAV permettant de bénéficier de fonctionnalités de stockage et d'accès aux fichiers dans des applications Java.

WebDAV (*Web-based Distributed Authoring and Versioning*) est une extension du protocole HTTP, développée à l'origine (en 1996) pour que des utilisateurs puissent ajouter, éditer et supprimer des fichiers sur des serveurs Web distants. Le protocole WebDAV gère par ailleurs les accès concurrents aux fichiers de manière à éviter, par exemple, les écritures sur un fichier pendant que celui-ci est lu.

L'avantage principal d'utilisation d'un serveur WebDAV par rapport à un stockage simple de fichiers dans un dossier est la gestion des accès concurrents qu'offre WebDAV. Le second avantage est que le serveur de fichiers est indépendant du serveur de sécurité. Il serait donc possible d'utiliser deux serveurs physiques distincts, l'un pour la gestion de la sécurité, le second pour le stockage des fichiers.

Il est à noter à titre d'information que le projet Slide a très récemment été abandonné par Jakarta (le 3 novembre 2007) au profit d'une nouvelle solution nommée Apache Jackrabbit. [10] [11]

9.2.4 Serveur Web

Le serveur Web utilisé dans ce projet est basé sur la solution Apache Tomcat développée par la fondation Apache Software Foundation.

Tomcat était à la base un serveur indépendant développé par Jakarta et implémentant les spécifications des servlets et des JSP uniquement, puis ce serveur a été intégré au serveur Web Apache afin de supporter les pages Web traditionnelles. Apache est le serveur Web le plus répandu sur Internet actuellement, avec un taux d'utilisation de 47.73% (octobre 2007). [12]

Ce serveur Web offre ainsi les fonctionnalités correspondant parfaitement aux exigences de ce projet. Il offre par ailleurs aussi un système d'ajout et de suppression d'applications Web simple d'utilisation et performant. Enfin, ce serveur est gratuit d'utilisation. La version utilisée dans ce projet est la version 5 intégrant directement le serveur WebDAV, facilitant ainsi la mise en place de ce serveur. [13]

10. Service d'authentification – fonctionnement théorique

Le service à développer doit posséder un service d'authentification forte pour sécuriser l'accès aux données confidentielles.

Une authentification forte permet de contrôler au minimum deux des éléments suivants : ce que la personne sait, ce que la personne possède et ce que la personne est. Chaque type d'authentification possède des avantages et des inconvénients. En effectuant une authentification forte, il est alors possible de profiter des multiples avantages de chaque type d'authentification, tout en réduisant les inconvénients de ceux-ci.

Dans le cadre de ce projet, trois types d'authentification différents ont été définis : l'authentification formelle (correspond à ce que la personne sait), l'authentification matérielle (correspond à ce que la personne possède) et l'authentification externe qui peut être une combinaison de ce que l'utilisateur possède et/ou de ce que l'utilisateur est.

Ces différents types d'authentification sont présentés de manière détaillée ci-dessous.

10.1 Authentification formelle

Ce type d'authentification se base sur un mot de passe textuel simple et offre une méthode d'authentification relativement sûre du moment que l'utilisateur est le seul à connaître le mot de passe et que ce dernier soit assez complexe pour ne pas pouvoir être deviné facilement.

10.1.1 Avantage

- ◇ Protection efficace.

10.1.2 Inconvénients

- ◇ Temps de saisie élevé, en particulier sur des systèmes mobiles à l'interface de saisie restreinte.
- ◇ Le mot de passe nécessite d'être changé régulièrement afin de garder un bon niveau de protection.
- ◇ Le mot de passe peut être usurpé à l'utilisateur.

10.2 Authentification matérielle

Cette méthode d'authentification utilise un identifiant matériel qui est lié à l'appareil mobile de l'utilisateur (cartes SIM, certificats stockés sur des périphériques amovibles, etc). Ces identifiants matériels permettent d'authentifier l'appareil utilisé, et ce, afin de pouvoir ou non lui accorder l'accès aux ressources demandées.

10.2.1 Avantages

- ◇ Aucun échange d'informations n'est nécessaire avec l'utilisateur, cette authentification est donc transparente pour ce dernier.
- ◇ Authentification relativement rapide.

10.2.2 Inconvénients

- ◇ L'appareil peut être volé, donc l'identifiant matériel de l'appareil aussi.

- ◇ Cette méthode authentifie bien l'appareil, mais n'authentifie pas l'utilisateur de l'appareil, ce système d'authentification ne peut donc pas être utilisé seul, car toute personne en possession de l'appareil pourrait accéder aux données librement si l'appareil ne dispose d'aucune autre méthode de protection.

10.3 Authentification externe

L'authentification externe permet d'ajouter un troisième type d'authentification afin de compléter l'authentification formelle et l'authentification matérielle. Le terme « authentification externe » signifie que l'on authentifie l'utilisateur en contrôlant sa présence. Cela permet de s'assurer que c'est bien l'utilisateur présent qui utilise l'appareil mobile et non une autre personne.

Ce contrôle peut être effectué à l'aide d'une authentification biométrique ou d'un autre type d'authentification permettant de localiser l'utilisateur et l'appareil à protéger.

10.3.1 Authentification biométrique

L'authentification biométrique se base sur les caractéristiques physiques de l'utilisateur pour pouvoir l'authentifier. Parmi les principales possibilités d'authentifications biométriques, l'analyse de la main (empreintes digitales ou silhouette de la main), l'analyse de l'œil de l'utilisateur (iris ou rétine), l'analyse du visage ou encore de la voix de l'utilisateur sont les méthodes principalement utilisées actuellement. La puissance d'analyse nécessaire (coût) et la précision de l'authentification sont les deux paramètres importants permettant de classer ces différentes technologies, comme le montre la figure 6 : Classification des méthodes d'authentification biométrique [14].

10.3.1.1 Avantages de l'authentification biométrique

- ◇ Authentification rapide.
- ◇ Falsification difficile.

10.3.1.2 Inconvénients de l'authentification biométrique

- ◇ Le taux d'erreurs lors de l'authentification n'est pas nul, surtout dans les méthodes d'authentification encore expérimentales. La précision de l'authentification est donc un facteur très important et une authentification moyennement précise ne devrait être utilisée qu'en collaboration avec d'autres méthodes d'authentification.
- ◇ Les scanners, lecteurs ou autres dispositifs d'authentification, ainsi que les pilotes ou logiciels les accompagnant, sont actuellement encore chers en raison de leur diffusion restreinte.
- ◇ Les méthodes d'authentification sont souvent sensibles aux conditions dans lesquelles l'utilisateur se trouve (pour une authentification du visage, l'éclairage extérieur est un facteur important, ainsi que la position de la tête). Le taux d'erreurs lors d'authentifications biométriques est donc plus élevé que pour d'autres types d'authentification.

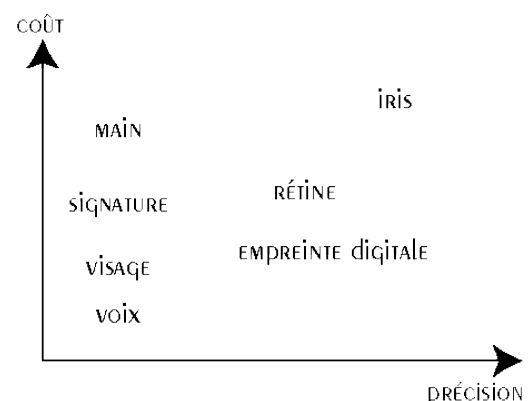


Figure 6: Classification des méthodes d'authentification biométrique

10.3.2 Authentification géographique

Cette authentification permet de localiser géographiquement à la fois l'utilisateur et son appareil mobile

et de contrôler la distance qui les sépare. Lorsque la distance devient trop importante (en cas de vol par exemple), un verrouillage de l'appareil mobile est nécessaire.

Une telle authentification pourrait s'effectuer à l'aide d'un badge contenant un module Bluetooth et qui serait connecté à l'appareil mobile. Ce badge serait alors porté par l'utilisateur sur sa veste ou dans sa poche. En cas de vol de l'appareil mobile, celui-ci serait éloigné du badge engendrant une rupture de la connexion. Une détection du vol serait alors possible, et donc le blocage automatique de l'accès à l'appareil mobile.

On peut aussi imaginer une localisation à l'aide d'un module GPS permettant de mesurer très précisément la distance entre l'utilisateur (portant un badge GPS) et l'appareil mobile.

10.3.2.1 Avantages de l'authentification géographique

- ◇ Authentification rapide.
- ◇ Falsification difficile.
- ◇ Aucune interaction avec l'utilisateur n'est nécessaire.

10.3.2.2 Inconvénients de l'authentification géographique

- ◇ Erreur de mesure possible (interférences électromagnétiques avec la technologie Bluetooth empêchant la connexion entre le badge et l'appareil mobile ou perte de connexion GPS à l'intérieur de bâtiments).

10.4 Gestion des méthodes d'authentification

Les demandes d'authentification se feront de manière à minimiser les interactions avec l'utilisateur, tout en essayant de garder un haut niveau de sécurité. Les différentes situations possibles sont résumées dans le tableau du chapitre 10.4.7, page 29, les détails étant donnés ci-dessous.

10.4.1 Démarrage du client lourd

L'exécution du programme s'effectue habituellement en début de service pour le médecin. Aucune urgence n'a à être traitée à ce moment-là, une authentification plus sûre et nécessitant donc un peu plus d'interaction avec l'utilisateur peut facilement être effectuée.

Dans ce cas-là, une fois que le médecin s'est identifié, les méthodes d'authentification suivantes devraient être effectuées :

- ◇ Authentification formelle
- ◇ Authentification matérielle

Ces authentifications permettent, d'une part, d'authentifier l'appareil afin de lui accorder ou de lui refuser l'accès au service, et d'autre part d'authentifier l'utilisateur afin de savoir quels sont ses droits et ainsi à quelles applications il a le droit d'accéder.

10.4.2 Contrôles périodiques au niveau des applications

Lors du service du médecin, il est envisageable qu'il laisse l'appareil mobile sans surveillance pendant un certain temps. Durant cette période, il est possible que l'appareil mobile soit utilisé ou volé par une tierce personne. Il se peut aussi que l'appareil tombe de la poche du médecin sans que celui-ci s'en rende compte. Les accès au service d'informations médicales par une personne non autorisée n'étant

pas souhaités, il est nécessaire de protéger l'accès au service périodiquement en effectuant une demande d'authentification.

Toutefois, le délai entre les demandes d'authentifications étant relativement court, une méthode d'authentification nécessitant peu d'interaction avec l'utilisateur, mais qui permette tout de même de vérifier l'identité du médecin, est donc nécessaire.

Dans ce cas-là, la méthode d'authentification la plus appropriée à cette situation serait la suivante :

◇ Authentification externe

Cette authentification serait demandée toutes les 15 minutes (cette durée sera paramétrable), peu importe l'application utilisée. Ceci permet de limiter l'accès à 15 minutes au maximum si l'appareil mobile a été volé, ensuite de quoi tout le service est bloqué et demande une nouvelle authentification.

Cette authentification sera par ailleurs aussi demandée lors du démarrage de nouvelles applications. Cette seconde contrainte permet, si une personne non autorisée utilise l'appareil mobile, de ne pas avoir la possibilité d'ouvrir de nouvelles applications. Une telle personne serait donc limitée aux applications déjà ouvertes.

En revanche, le médecin doit parfois ouvrir plusieurs applications dans un court laps de temps, il faut alors éviter de lui demander de s'authentifier à chaque ouverture d'application. Ceci peut être fait en introduisant un délai durant lequel le médecin peut ouvrir des applications sans demandes d'authentifications. Ce délai serait typiquement de 5 minutes et débiterait après une authentification externe, formelle ou matérielle. Une fois le délai échu, une nouvelle demande serait effectuée si une application devait être ouverte.

Ces différents délais permettent ainsi de s'assurer qu'une personne malintentionnée ne puisse pas utiliser les applications déjà ouvertes pour une durée supérieure à 15 minutes et qu'elle ne puisse pas démarrer de nouvelles applications 5 minutes après la dernière authentification externe, matérielle ou formelle effectuée par le médecin.

10.4.3 Contrôles périodiques longs

En dehors des contrôles périodiques fixes, il est important d'effectuer des contrôles plus stricts après une certaine période prolongée. En effet, certains appareils mobiles ne ferment pas les applications qui sont en mémoire lors de leur extinction (ou mise en veille). De ce fait, le contrôle de démarrage de l'application ne serait effectué qu'une seule fois si l'application n'est jamais fermée. Il est donc important d'effectuer une nouvelle identification et authentification de l'utilisateur et de l'appareil régulièrement afin de s'assurer de leur validité. Ceci est le but des contrôles périodiques longs.

Ainsi, toutes les 4 heures, le système effectue une demande d'authentification similaire à l'authentification de démarrage (c'est-à-dire par mot de passe et matérielle).

Un tel système, bien qu'efficace, empêcherait le médecin de pouvoir rapidement répondre à une alerte si elle se déclarait au milieu de la nuit par exemple. Il faut donc prévoir ce type d'alertes et la possibilité de contourner temporairement l'authentification formelle « lourde » afin de pouvoir répondre à l'alerte. Dans ce cas-là, le médecin doit pouvoir choisir de s'authentifier avec l'authentification externe, ce qui lui donnerait accès au service pour un certain temps (typiquement 5 minutes). Ensuite, le service demanderait à nouveau l'authentification formelle, qui ne pourrait alors plus être contournée. Ce contournement doit par ailleurs pouvoir être désactivé par l'administrateur du service si tel est son souhait, en raison de la légère diminution de sécurité qu'un tel évitement peut engendrer.

En résumé, après une période prolongée (4 heures), le service requiert :

◇ Authentification formelle – peut être contournée pour une durée de 5 minutes par

l'authentification externe

- ◇ Authentification matérielle

10.4.4 *Echecs d'authentifications*

Un échec d'authentification peut survenir de temps à autre, en raison d'une erreur de frappe pour l'authentification formelle ou une erreur de lecture lors de l'authentification externe (dans le cas d'une authentification biométrique par exemple). En revanche, un trop grand nombre d'échecs pourrait signifier qu'une personne est en train d'essayer de trouver le mot de passe par force brute. De ce fait, il faut configurer des seuils de limite pour le nombre d'authentifications erronées. Ces seuils sont donnés ci-dessous :

- ◇ Authentification formelle :
 - Trois erreurs maximales. Si ce seuil est dépassé, l'accès au service pour cet utilisateur est bloqué et doit être débloqué par l'administrateur du service⁴.
- ◇ Authentification matérielle :
 - Aucune erreur n'est tolérée ici. Si l'appareil mobile échoue l'authentification matérielle, cela signifie que l'appareil n'est pas autorisé à utiliser ce service.
- ◇ Authentification externe :
 - L'authentification externe n'étant pas une authentification extrêmement précise (si l'authentification biométrique est utilisée par exemple), il se peut qu'elle génère quelques authentifications erronées. Un seuil de cinq authentifications erronées est donc raisonnable. Si ce seuil est dépassé, l'accès au service pour cet utilisateur est bloqué et doit être débloqué par l'administrateur du service.

Les seuils d'échecs d'authentifications des authentifications formelles et externes peuvent être paramétrés par l'administrateur indépendamment pour chaque groupe d'utilisateurs⁵.

10.4.5 *Désactivation de l'authentification externe*

Dans le cas où l'appareil mobile ne dispose pas d'un système d'authentification externe, toutes les authentifications externes sont remplacées par des authentifications formelles. Il n'est alors plus possible d'éviter une authentification formelle, car cela engendrerait une seconde authentification formelle (remplaçant l'authentification externe).

Ce choix pourra être paramétré par l'administrateur du service.⁶

10.4.6 *Désactivation de l'évitement d'authentification formelle*

Le contournement temporaire de l'authentification formelle peut également être désactivé. Ce choix pourra être paramétré par l'administrateur du service.⁷

⁴ Pour plus d'informations, voir le chapitre 19.3.3, page 101

⁵ Pour plus d'informations, voir le chapitre 19.3.5, page 102

⁶ Pour plus d'informations, voir le chapitre 19.3.6, page 103

⁷ Pour plus d'informations, voir le chapitre 19.3.5, page 102

10.4.7 Récapitulatif des règles d'authentification

La description détaillée des différentes authentifications peut être résumée de la manière suivante :

Type d'authentification	Moment de la demande	Résumé	Remarques
Authentification formelle (Trois authentifications erronées maximum)	Au démarrage du client lourd	Permet l'authentification de l'utilisateur.	
	Toutes les 4 heures	Protection lors de la non-fermeture du client lourd afin d'éviter que cette authentification ne soit faite qu'une seule fois. Cette durée est paramétrable par l'administrateur du service.	Peut être contournée pour une durée de 5 minutes avec une authentification externe, en cas d'urgence par exemple.
	Au moment des authentifications externes, si une telle authentification n'est pas disponible sur l'appareil mobile	Si l'appareil ne dispose pas de système d'authentification externe, ce type d'authentification est remplacé par des authentifications formelles.	Si l'authentification externe est désactivée, il n'est plus possible d'éviter une authentification formelle.
Authentification matérielle (Une authentification erronée maximum)	Au démarrage du client lourd	Permet l'authentification de l'appareil mobile.	
	Toutes les 4 heures	Protection lors de la non-fermeture du client lourd. Cette durée est paramétrable par l'administrateur du service.	
Authentification externe, par exemple biométrique (Cinq authentifications erronées maximum)	Lors du démarrage d'une application	Authentification demandée lors du premier démarrage d'une application afin d'éviter qu'une personne malveillante en possession de l'appareil ne puisse démarrer de nouvelles applications.	L'authentification n'est pas redemandée pendant les 5 minutes suivant le premier démarrage d'une application, puis une nouvelle demande est formulée si une application doit être ouverte après ce délai.
	Toutes les 15 minutes	Demande d'authentification externe toutes les 15 minutes afin qu'une personne malintentionnée ne puisse utiliser le service que pour une durée maximale de 15 minutes. Cette durée est paramétrable par l'administrateur du service.	Le temporisateur est remis à sa valeur initiale pour toute authentification qui aurait été faite durant l'intervalle des 15 minutes entre deux authentifications externes.

Remarque :

Ces différents délais sont donnés à titre informatif et peuvent tous être modifiés par l'administrateur du service afin de mieux correspondre aux critères de sécurité souhaités. Il est à noter que les délais d'authentifications externes donnés ci-dessus sont adaptés à une authentification biométrique requérant une interaction avec l'utilisateur. Dans le cas d'utilisation d'une authentification externe nécessitant moins d'interaction, il est conseillé de réduire le délai de 15 minutes à 2 minutes par exemple.

10.5 Exemples de scénarios d'utilisation

Afin d'illustrer les règles d'authentification, deux exemples sont présentés ci-dessous, l'un lors d'une utilisation normale, l'autre dans une situation plus problématique. L'authentification externe est ici de type biométrique.

10.5.1 Utilisation normale

- ◇ 8h00 du matin
 - Le médecin allume son appareil mobile et ouvre le client lourd (bureau mobile). Dès l'ouverture du programme, l'authentification matérielle est activée afin d'authentifier l'appareil mobile. Une fois l'appareil vérifié et validé, une authentification formelle est demandée au médecin. Celui-ci saisit son mot de passe, puis valide la saisie. Le médecin peut alors ouvrir d'autres applications. Imaginons que nous avons à disposition les applications A, B, C et D pour ce médecin. Il ouvre l'application A et commence à l'utiliser.
- ◇ 8h10
 - Le médecin souhaite ouvrir l'application B. Une authentification biométrique est requise, car l'ouverture de l'application B s'est faite après les 5 minutes suivant l'ouverture de l'application A.
- ◇ 8h25
 - Une authentification biométrique est requise afin de contrôler que ce soit bien le médecin qui utilise l'appareil mobile, ce qui est le cas ici.
- ◇ 8h30
 - Le médecin a terminé de travailler sur l'appareil mobile et le met dans sa poche.
- ◇ 9h00
 - Une alerte est déclenchée, et l'appareil mobile le signale. Le médecin doit s'authentifier de manière biométrique avant de pouvoir traiter l'alerte.
- ◇ 9h10
 - L'alerte est traitée et le médecin remet l'appareil dans sa poche.
- ◇ 12h00
 - Une authentification matérielle est automatiquement effectuée, puis une authentification formelle est affichée à l'écran. Le médecin ne sort cependant pas l'appareil de sa poche.
- ◇ 15h00
 - Une alerte est déclenchée, le médecin est toujours invité à s'authentifier de manière formelle. Cependant, au vu de l'urgence, le médecin sélectionne de contourner l'authentification formelle en utilisant l'authentification biométrique et s'authentifie de cette manière. L'urgence nécessite l'appel d'une ambulance, le médecin appelle donc les urgences.
- ◇ 15h05
 - Le service est bloqué, le médecin est invité à s'authentifier de manière formelle pour pouvoir continuer à utiliser le service. Le médecin n'a plus besoin de s'authentifier, l'alerte est terminée, il remet donc l'appareil dans sa poche jusqu'à la fin de son service.

10.5.2 Utilisation problématique

- ◇ 8h00 du matin
 - Le médecin allume son appareil mobile et ouvre le client lourd (bureau mobile). Dès l'ouverture du programme, l'authentification matérielle est effectuée afin d'authentifier l'appareil mobile. Une fois l'appareil vérifié et validé, une authentification formelle est demandée au médecin. Celui-ci saisit son mot de passe, puis valide la saisie. Imaginons que nous avons à disposition les applications A, B, C et D pour ce médecin. Il ouvre l'application A et commence à l'utiliser.
- ◇ 8h05
 - Le médecin met l'appareil mobile dans sa poche, car il a terminé de l'utiliser.
- ◇ 8h07
 - L'appareil mobile tombe de la poche du médecin, qui ne le remarque pas.
- ◇ 8h10
 - Un inconnu ramasse l'appareil mobile et essaie de l'utiliser. Il voit que l'application A est ouverte et il voit qu'il est possible d'ouvrir d'autres applications. Il essaie d'ouvrir l'application B mais une invitation à une authentification biométrique lui est soumise. L'inconnu met l'appareil mobile dans sa poche.
- ◇ 9h40
 - Une alerte est déclenchée et l'appareil mobile le signale. L'inconnu essaie de s'authentifier de manière biométrique, mais échoue les 5 authentifications. Le service est bloqué et l'inconnu décide de retourner l'appareil mobile au secrétariat afin qu'il soit restitué au médecin.
- ◇ 11h30
 - Le médecin récupère l'appareil mobile. Il va voir le responsable du service afin de lui expliquer la situation pour que son compte soit à nouveau débloqué. Le médecin doit alors redémarrer l'application et recommencer les étapes d'authentifications.
- ◇ 12h50
 - Le médecin veut consulter quelques fiches de patients et s'authentifie de manière biométrique, puis il pose l'appareil mobile sur une table.
- ◇ 12h53
 - Dans un moment d'inattention du médecin, une personne vole l'appareil mobile. Cette personne ouvre rapidement alors les applications B et C.
- ◇ 12h56
 - L'inconnu veut ouvrir l'application D, mais une authentification biométrique est requise, car les cinq minutes entre l'ouverture de la première application et l'ouverture de cette application sont passées. Face à cette demande, l'inconnu repose l'appareil mobile sur l'une des tables et s'en va.

11. Service d'authentification – réalisation pratique

Cette partie explique comment le service de gestion des authentifications a été implémenté dans le client lourd.

11.1 Service de gestion des authentifications

En se référant au tableau «Récapitulatif des règles d'authentification» du chapitre 10.4.7 à la page 29, il a été possible de définir quatre durées principales pour la gestion des demandes d'authentifications :

- ◇ Délai de démarrage d'applications : ce délai représente le temps écoulé entre le démarrage de la première application depuis que ce compteur de durée a été initialisé jusqu'au démarrage de la dernière application dans cet intervalle de temps. Si une application démarre après ce délai, une nouvelle authentification externe est requise.
- ◇ Délai d'évitement d'authentification : ce délai représente le temps écoulé entre l'évitement d'une authentification formelle afin de pouvoir s'authentifier de manière externe jusqu'à l'obligation d'effectuer à nouveau une authentification formelle pour compenser l'évitement.
- ◇ Long délai périodique d'authentification : ce délai permet d'effectuer périodiquement une authentification formelle et matérielle.
- ◇ Court délai périodique d'authentification : ce délai permet d'effectuer périodiquement une authentification externe.

Les authentifications périodiques (courtes ou longues) doivent être exécutées automatiquement par le client lourd, peu importe l'état des applications en cours d'exécution. Il en va de même pour l'authentification formelle lors d'un évitement d'authentification. Pour ce faire, un processus vérifie chaque seconde si l'un de ces trois délais est échu, et si tel est le cas, effectue les authentifications nécessaires.

Le délai de démarrage des applications n'est en revanche pas géré de la même manière. Lorsqu'une application désire envoyer une requête vers un serveur, elle doit préalablement obtenir l'identifiant de sécurité d'applications contenu dans le client lourd. Cet identifiant, une fois intégré dans la requête, permet aux serveurs d'applications de s'assurer que l'application est autorisée à effectuer la requête⁸. C'est durant l'obtention de cet identifiant qu'un contrôle s'effectue sur l'application en déclenchant une demande d'authentification si nécessaire. Le contrôle d'une application est réalisé en trois étapes :

1. Contrôle si l'application qui démarre est une nouvelle application. Ce contrôle s'effectue en vérifiant si son nom apparaît dans une liste locale des applications en exécution. Si tel est le cas, alors l'application est autorisée et aucun des contrôles supplémentaires ci-dessous n'est effectué. Sinon, les contrôles suivants doivent être effectués.
2. Contrôle si le délai de démarrage des applications est échu. Si tel est le cas, alors une demande d'authentification externe est effectuée.
3. Contrôle sur le serveur si l'utilisateur est autorisé à utiliser l'application. Si tel est le cas, alors le nom de l'application est ajouté dans la liste des applications en exécution afin d'éviter les contrôles supplémentaires lors de la prochaine requête⁹. En revanche, si l'application n'est pas autorisée, alors aucun identifiant de sécurité n'est retourné à cette application. Dans ce cas, tous ces contrôles s'effectueront à nouveau si une nouvelle requête est effectuée

⁸ L'explication détaillée du fonctionnement de cet identifiant est donnée au chapitre 13.3, à la page 54.

⁹ Une application qui est fermée et ouverte à nouveau n'est pas considérée comme une nouvelle application, à moins qu'elle ait été supprimée de la liste des applications en exécution suite à une authentification et qu'elle ait été rouverte après le délai de démarrage des applications.

ultérieurement par la même application.

Il faut encore préciser qu'une seule authentification peut être exécutée simultanément. Cela induit que si une application désire obtenir l'identifiant de sécurité pendant qu'une authentification est en cours ou en attente, alors l'obtention de cet identifiant est mise en attente jusqu'à l'aboutissement de l'authentification. Comme les identifiants de sécurité sont modifiés régulièrement (après chaque authentification formelle), cette mise en attente permet d'éviter d'obtenir des identifiants obsolètes ou invalides.

Remarque :

Une application a le choix de ne pas utiliser l'identifiant de sécurité dans des cas exceptionnels, ce qui enlèverait alors toute garantie de sécurité de l'application. Ce choix peut toutefois être utile pour des applications prioritaires, telles que les contrôles de présence d'alertes où aucune donnée confidentielle n'est transmise, afin que ces demandes ne soient pas mises en attente lorsque des authentifications sont en cours.

11.1.1 Remise à zéro des délais et de la liste des applications en exécution

Les délais et la liste des applications en cours d'exécution sont remis à zéro dans les différents cas décrits ci-dessous :

- ◇ Le délai périodique long est remis à zéro :
 - Après une authentification découlant d'un contrôle périodique long
 - Après une authentification découlant d'un évitement d'authentification formelle
- ◇ Le délai périodique court est remis à zéro :
 - Après tous les types d'authentifications
- ◇ Le délai de démarrage des applications est remis à zéro :
 - Après tous les types d'authentifications
- ◇ Le contrôle des applications s'effectue à nouveau :
 - Après une authentification découlant d'un contrôle périodique long ou court
 - Après une authentification découlant du démarrage d'une nouvelle application effectué après le délai de démarrage des applications
- ◇ Le délai d'évitement d'authentification formelle est supprimé :
 - Après une authentification découlant d'un évitement d'authentification formelle

Remarque :

Les authentifications découlant d'un contrôle périodique court et d'un contrôle d'application effectuent des opérations très semblables. Cela s'explique par le fait que si aucune application n'est démarrée ou si une application n'envoie pas de requêtes au serveur, des contrôles périodiques doivent être effectués afin de garantir la sécurité d'accès à l'appareil mobile. En revanche, si de nouvelles applications sont ouvertes régulièrement, alors les authentifications externes découlant des contrôles des applications sont suffisantes et ne requièrent pas d'authentifications périodiques externes supplémentaires. C'est pourquoi ces deux types d'authentification réinitialisent les mêmes délais.

Enfin, la liste contenant les noms des applications démarrées est réinitialisée après chaque authentification dans le but d'obliger le client à renouveler régulièrement les contrôles de chaque application.

11.1.2 Illustrations de la gestion des demandes d'authentifications

La figure 7 «Diagramme temporel de la gestion des authentifications en situation normale» indique le déroulement des demandes d'authentification dans le cas suivant :

- ◇ Délai périodique court : 60 secondes
- ◇ Délai périodique long : 130 secondes
- ◇ Délai d'évitement des authentifications : 30 secondes
- ◇ Aucune application n'est démarrée après le démarrage du gestionnaire d'authentifications
- ◇ L'authentification externe est ici une authentification biométrique, à titre d'exemple.

Dans l'exemple de la figure 7, il est possible d'observer que la première authentification biométrique s'effectue après 60 secondes. L'authentification elle-même requiert 6 secondes (temps estimé), puis le compteur est remis à zéro. Soixante secondes plus tard, une nouvelle authentification biométrique est requise, durant elle aussi 6 secondes. Au total, 132 secondes se sont écoulées depuis le début. Comme une seule authentification peut se faire en même temps, l'authentification matérielle et formelle qui auraient dû se déclencher après 130 secondes sont effectuées après l'authentification biométrique.

Il est à noter que de tels délais ne sont donnés qu'à titre d'exemple et afin de pouvoir effectuer des vérifications d'exactitude en pratique. Ils seraient toutefois bien trop courts en utilisation normale.

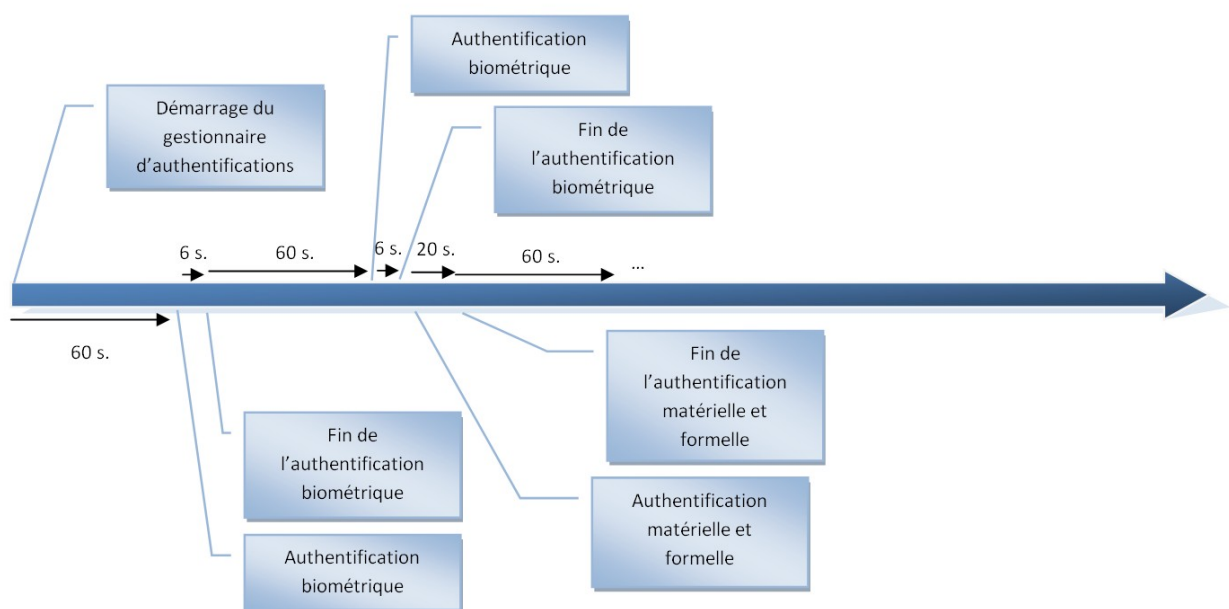


Figure 7: Diagramme temporel de la gestion des authentifications en situation normale

La figure 8 représente la même situation que la figure précédente, mais l'authentification formelle est évitée par l'utilisateur :

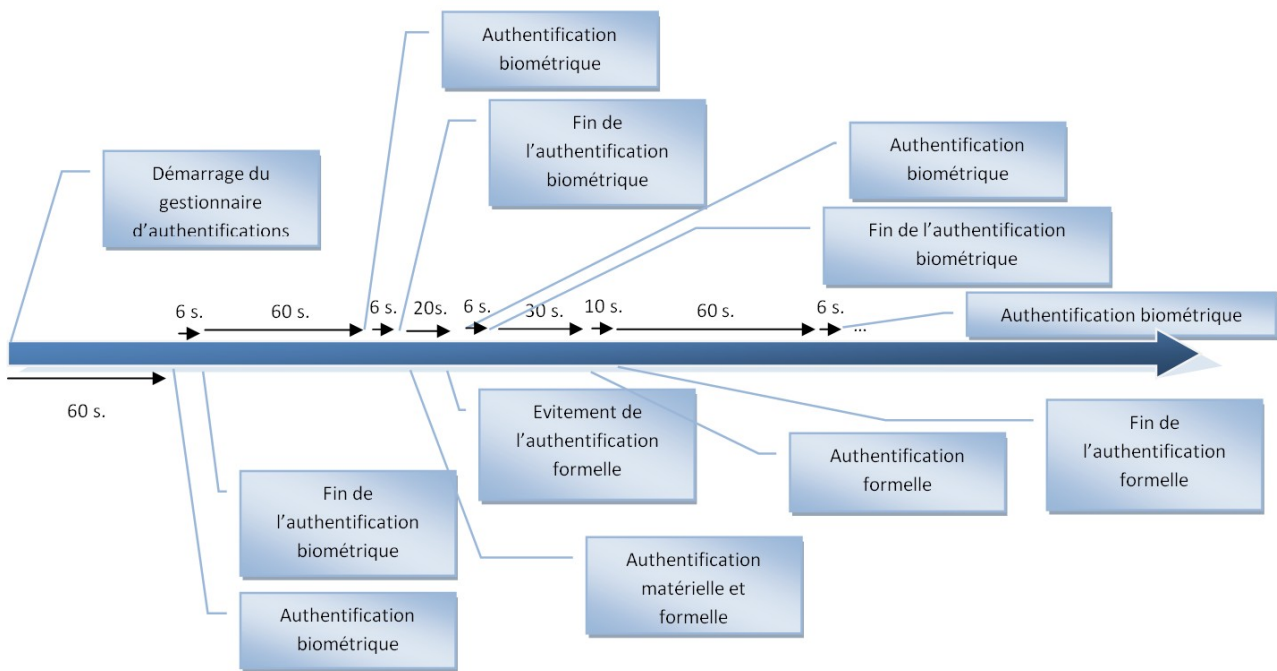


Figure 8: Diagramme temporel de la gestion des authentifications en situation d'évitement

Il est possible de remarquer que la figure est identique jusqu'à la première authentification formelle. A ce moment-là, l'utilisateur décide d'éviter l'authentification formelle. Cet évitement engendre la demande d'une authentification biométrique qui dure 6 secondes. Un délai d'évitement se déclenche dès la fin de l'authentification biométrique et effectue une demande d'authentification formelle non évitable 30 secondes plus tard.

Ici aussi, les délais utilisés ne sont donnés qu'à titre d'exemples et de vérifications pratiques, ceux-ci étant bien trop courts pour être utilisés en réalité.

11.2 Implémentation du service d'authentification formelle

Le service d'authentification formelle s'occupe de demander le nom de l'utilisateur (ou plus communément appelé *login*) et son mot de passe dans le but de pouvoir authentifier l'utilisateur.

La principale difficulté est d'adapter l'interface de saisie afin de minimiser les interactions de l'utilisateur avec l'appareil mobile. La solution retenue pour respecter cette contrainte est de proposer à l'utilisateur une liste contenant le nom de tous les utilisateurs autorisés à s'authentifier sur l'appareil mobile utilisé. De cette manière, l'utilisateur n'aura qu'à sélectionner son nom dans la liste et saisir son mot de passe afin de s'authentifier.

Bien que l'affichage de ces noms puisse éventuellement être une source d'informations pour une personne malveillante, les noms des utilisateurs ne sont généralement pas critiques et peuvent être trouvés d'une autre manière très aisément, en étudiant la structure de la composition de ces noms par exemple. En effet, chaque entreprise possède une méthode pour générer les noms d'utilisateurs, en prenant par exemple la première lettre du prénom suivi du nom de famille.

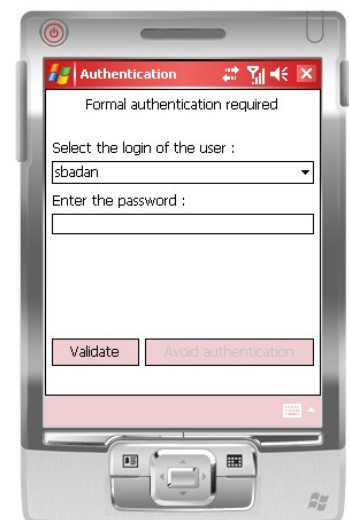


Figure 9: Fenêtre d'authentification formelle

11.3 Implémentation du service d'authentification matérielle

11.3.1 Fonctionnement théorique

Pour rappel, ce service doit permettre d'authentifier l'appareil mobile. Il existe plusieurs solutions pour effectuer cela. L'une des solutions possibles serait la mise en place d'un système de certificats entre l'appareil mobile et le serveur. Dans ce cas-là, seuls les appareils ayant le bon certificat installé seraient en mesure de s'authentifier sur le serveur. Cette solution pose en revanche deux problèmes principaux :

1. Le certificat doit être créé et stocké sur l'appareil mobile. Il faut donc trouver un moyen sûr de protection du certificat contre la lecture sur l'appareil mobile afin d'éviter qu'il soit copié et installé sur un autre appareil mobile.
2. Le certificat n'est pas lié au matériel. Bien que le certificat soit une bonne protection pour l'authentification de l'appareil, il n'offre aucune protection sur la copie intégrale de l'OS de l'appareil mobile (et donc de ses certificats) dans le but de l'installer sur un autre appareil mobile similaire. Une telle copie n'est pas difficile à effectuer sur un tel appareil, particulièrement en raison de la taille restreinte de l'OS et donc la rapidité de la copie.

L'utilisation de certificats offrirait donc une assez bonne sécurité, mais ils ne sont pas assez liés au matériel pour pouvoir authentifier l'appareil mobile de manière sûre.

Une autre solution consiste à utiliser un identificateur unique afin d'identifier et d'authentifier l'appareil mobile. En partant du principe que tous les appareils mobiles auront en leur possession un module de téléphonie nommé GSM (*Global System for Mobile Communications*), il est possible d'utiliser le numéro IMEI (*International Mobile Equipment Identity*). L'IMEI est un numéro stocké dans le téléphone et l'identifie de manière unique, indépendamment de la carte SIM (*Subscriber Identity Module*). Ce numéro est conçu de manière à ne pas pouvoir être modifié facilement [15].

Le principe de cette solution est simple : au moment de l'authentification de l'appareil mobile, le client lourd va lire le numéro IMEI dans le téléphone et l'envoyer directement au serveur afin qu'il soit vérifié. S'il est accepté, le téléphone est authentifié.

L'avantage de cette solution est qu'elle est extrêmement simple dans le principe. De plus, même si une tierce personne connaissait ce numéro, elle serait incapable de le reproduire artificiellement au moment de l'authentification, car le client lourd va le lire directement au niveau du matériel. Cette solution permet donc d'authentifier le téléphone de manière simple, sûre et efficace.

Malheureusement, la manière de récupérer ces informations est actuellement totalement dépendante de l'appareil mobile utilisé. En effet, certains constructeurs ont prévu cette fonctionnalité et proposent des fonctions pour retrouver ce numéro. En revanche, d'autres constructeurs ne le permettent pas, ou alors en imposant de très lourdes contraintes, telles que l'obligation de posséder des certificats provenant des opérateurs téléphoniques afin de pouvoir exécuter les fonctions pour retrouver ces numéros. D'autre part, Java étant un langage de haut niveau, il lui est difficile d'accéder à ces informations qui sont situées proche du matériel.

Bien que Java ME CDC propose une méthode générique permettant de lire des propriétés propres au téléphone se nommant *getProperty(...)* et prenant en paramètre un *String* décrivant la propriété à lire, il n'est malheureusement pas possible de lire directement le numéro IMEI en raison d'une limitation au niveau de la machine virtuelle utilisée.

La lecture de ces numéros ne pose toutefois aucun problème sur Windows Mobile avec des applications spécifiquement conçues pour cette plateforme (en C++ par exemple). Il est ainsi possible de récupérer ces informations depuis une application Java, en utilisant JNI (*Java Native Interface*) qui appelle une librairie en C++ stockée sur l'appareil mobile retournant le code IMEI.

Il faut toutefois prendre en compte que l'ajout d'une librairie intermédiaire entre le client lourd et le matériel rend le système un peu moins sûr, en raison du remplacement possible de cette librairie par une librairie modifiée retournant un numéro falsifié. Un contrôle du code haché de la librairie externe permet de s'assurer de l'intégrité de la librairie avant la lecture, et donc de résoudre ce problème. Il en va de même du client lourd qui pourrait être décompilé de manière à désactiver les authentifications. Un contrôle du code haché du client lourd est donc aussi nécessaire, permettant d'améliorer la sécurité.

Le schéma 10, « Authentification matérielle sur Windows Mobile », montre le fonctionnement de cette solution sur Windows Mobile 5.0. Ce fonctionnement peut être résumé de la manière suivante :

1. L'authentification matérielle est appelée par le service de gestion des authentifications. Le service d'authentification appelle la librairie en C++ dans le but de lire les informations spécifiques au téléphone.
2. La librairie C++ lit les informations dans l'appareil mobile et les retourne à l'application Java. Ces informations sont :
 - Le nom du fabricant du téléphone
 - Le modèle du téléphone
 - Le numéro de révision du téléphone
 - Le code IMEI du téléphone
 - Le numéro IMSI (*International Mobile Station Identity*) de la carte SIM du téléphone
3. Le service d'authentification matérielle reçoit ces informations puis envoie le code IMEI et les différents codes hachés au serveur qui effectuera une comparaison afin d'authentifier l'appareil mobile.

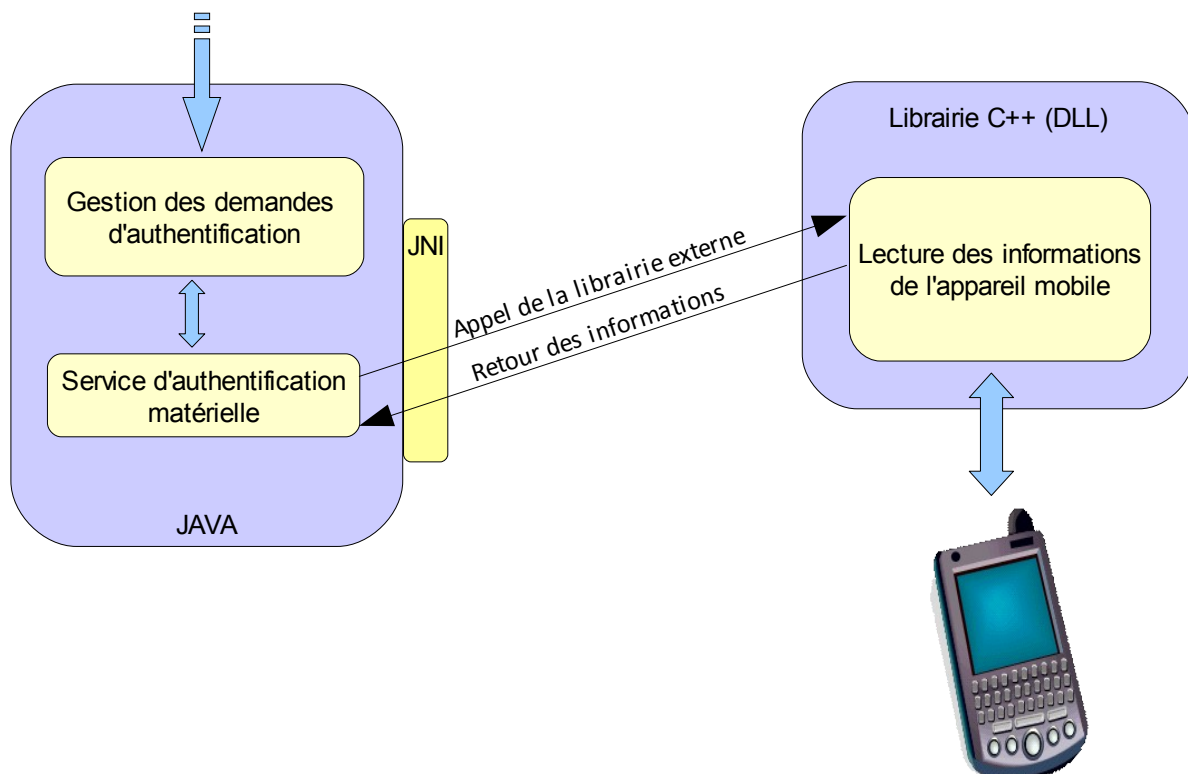


Figure 10: Authentification matérielle sur Windows Mobile

Dans le cadre de ce projet, l'authentification n'utilise que le code IMEI, sans le code IMSI qui permet d'identifier la carte SIM utilisée dans le téléphone. Cela permet au médecin de pouvoir changer la carte SIM sans avoir à demander à l'administrateur de modifier les accès au niveau du serveur, la carte SIM n'étant pas un élément critique au niveau de la sécurité du service. En outre, il est ainsi aussi possible d'utiliser l'appareil mobile sans carte SIM, si le médecin ne nécessite pas la fonction de téléphonie par exemple. L'on pourrait toutefois ajouter le contrôle du code IMSI ultérieurement très facilement si nécessaire.

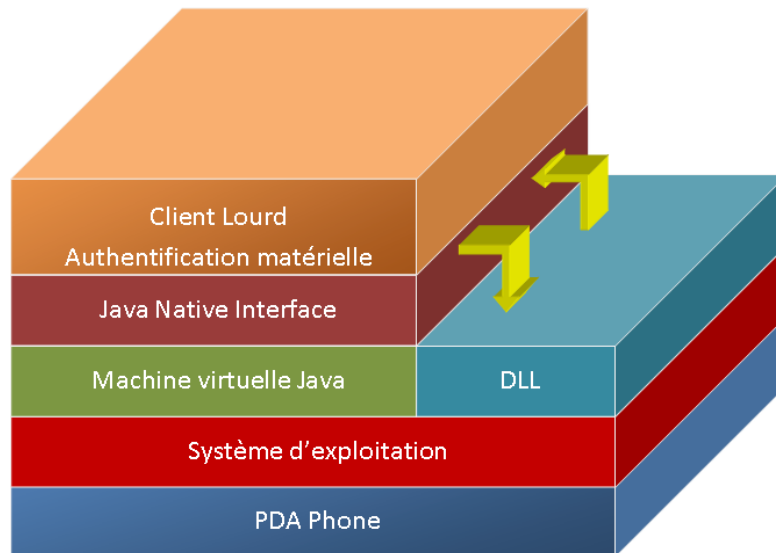


Figure 11: Représentation en couches de l'authentification matérielle

Au niveau des limitations des plates-formes, cette solution ne fonctionne que sur Windows Mobile 5.0, sur des appareils de type *PDA Phones* seulement.

En résumé, une authentification matérielle effectue les étapes suivantes :

1. Calcul du code haché de la librairie externe (DLL);
2. Calcul du code haché du client lourd;
3. Appel de la librairie externe par le programme Java;
4. Extraction des informations de l'appareil mobile par la librairie et retour de ces informations au programme Java;
5. Envoi du code haché du client, de la librairie et du code IMEI au serveur pour vérification de la validité de l'authentification;

11.3.2 Réalisation pratique

Au niveau pratique, les principales étapes pour créer la librairie et pour y accéder sont les suivantes :

1. Création de l'application Java en incluant une méthode de lecture avec le mot-clé *native*;
2. Appel à la librairie externe en ajoutant la méthode *System.load(...)* dans un bloc *static*. Il faut indiquer à cette méthode le chemin où sera placée la DLL afin que la machine virtuelle puisse la charger.
3. Compilation de la classe à l'aide de la commande *javac*. Il faut noter que même si la version Java utilisée sur le client lourd est la version Java ME CDC 1.1 équivalent à Java SE 1.4.2, il est tout à

fait possible d'utiliser une version plus récente de *javac* pour compiler la classe. En effet, la classe compilée n'est utilisée que pour générer l'en-tête C de la librairie. Ainsi, si la classe s'appelle *HelloWorld* et que cette classe appartient au paquetage *example*, la commande (dans le répertoire courant du projet) serait la suivante :

```
%JAVA_HOME%\bin\javac.exe -classpath example example\HelloWorld.java
```

4. Il faut ensuite créer l'en-tête du code C ou C++ en exécutant la commande *javah* sur la classe compilée :

```
%JAVA_HOME%\bin\javah.exe -classpath . example.HelloWorld
```

5. A l'aide de l'en-tête (fichier .h), il faut créer le corps du code (fichier .cpp), puis en générer une DLL. Pour ce faire, j'ai utilisé le programme *Microsoft Visual Studio 2005 Pro* sur lequel j'ai installé le kit de développement *Windows Mobile 5.0 SDK* [16]. Il faut alors créer un nouveau projet de type *C++\MFC Smart Device DLL*.
6. Dans le projet, il suffit ensuite de copier l'en-tête généré par la commande *javah*, puis de compléter le corps du programme.
7. La compilation et l'édition des liens ne peuvent s'effectuer que si les chemins d'accès vers les libraires JNI sont spécifiés dans la configuration du programme. Dans *Visual Studio*, il faut aller dans le menu *Tools/Options/Project and Solutions/VC++ Directories*, sélectionner la plateforme *Windows Mobile 5.0*, sélectionner *Include Files* puis ajouter les deux chemins suivants :

```
%JAVA_HOME%\include\win32
%JAVA_HOME%\include
```

où *%JAVA_HOME%* est le répertoire Java courant.

8. Une fois compilée, la DLL doit être placée dans le répertoire spécifié en 2.
9. Le programme est prêt à être exécuté;

En ce qui concerne la lecture des informations de l'appareil mobile, j'ai utilisé et adapté le code fourni dans les exemples du kit de développement *Windows Mobile 5.0* de Microsoft. L'exemple de base se nomme *ExTapi*, en raison de la librairie que ce programme utilise (*Extended Telephony Application Programming*). Cette librairie permet d'extraire diverses informations du téléphone, telles que le nom du fabricant, le modèle, le numéro de révision, le code IMEI ou encore le code IMSI du téléphone. J'ai adapté ce programme afin qu'il retourne les résultats sous forme d'une chaîne de caractères au travers de JNI. Le service d'authentification matériel Java peut ensuite extraire les informations souhaitées depuis la chaîne de caractères retournée.

Du point de vue de la licence sous laquelle le programme d'exemple a été fourni, celle-ci autorise de modifier, copier et de distribuer les programmes d'exemples si l'utilisateur accepte les termes de la licence MICROSOFT SOFTWARE LICENSE TERMS pour le kit de développement *Windows Mobile 5.0*. Cette licence est incluse dans le kit de développement et est affichée lors de l'installation de celui-ci.

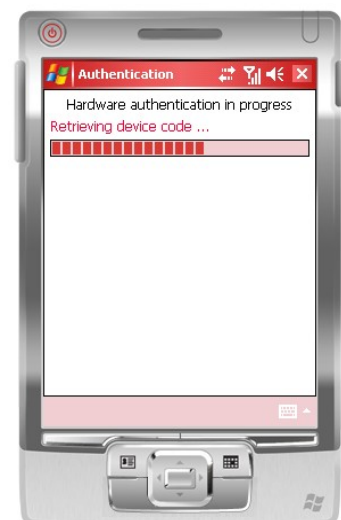


Figure 12: Fenêtre d'authentification matérielle

11.4 Implémentation du service d'authentification externe

Dans le cadre de ce projet, en particulier l'aspect lié à la mobilité du médecin, il est nécessaire de trouver une méthode d'authentification qui soit à la fois fiable et mobile. Ces deux contraintes font de l'authentification biométrique par empreinte digitale la meilleure solution pour cette situation.

Concernant les solutions existantes pour ce type d'authentification, en particulier les algorithmes de reconnaissance d'empreintes digitales, il n'existe actuellement que peu de produits, d'où un prix généralement élevé. Une possibilité serait de créer l'algorithme de vérification soi-même, mais l'implémentation d'un tel algorithme sortirait du cadre de ce projet en raison du temps requis pour une telle implémentation. Parmi les quelques solutions existantes, la solution qui s'adapterait le mieux à ce projet serait la suivante :

- ◇ GrFinger Java 4.5 est développée par la société Griaule. Cette solution inclut un SDK (Software Development Kit) qui permet de construire une application Java utilisant un service d'authentification biométrique. Ce service compare une empreinte digitale à une autre, puis en extrait un score qui détermine la correspondance entre les deux empreintes. Toute la gestion externe des empreintes (lecture, stockage, ...), qui sont stockées sous forme d'images, est à faire dans une application tierce, ce qui offre une grande souplesse d'intégration. Cette solution pourrait être utilisée au niveau du serveur, les images des empreintes digitales seraient alors envoyées des appareils mobiles après lecture de l'empreinte vers le serveur pour la vérification. Le prix de la licence pour cette solution est de US\$ 500, mais n'impose aucune limite quant au nombre d'utilisations de la licence. Une version de démonstration (contenant l'entier du SDK) est disponible pour 90 jours. [17]

Sun propose aussi une solution pour une authentification biométrique, reposant sur plusieurs de ses solutions de sécurité, telles qu'Access Manager (gestion centralisée de la sécurité pour des applications Web), JAAS (*Java Authentication and Authorization Service*) et SSO (Single Sign-On – permettant de s'authentifier une seule fois pour toutes les applications), qui sont toutes gratuites [18]. Ces services de SUN permettent de mettre en place l'infrastructure nécessaire à un service d'authentification spécifique, tel que le serveur BiObex [19] qui gère les authentifications par empreintes digitales. Malheureusement, le serveur BiObex est une solution destinée aux grandes entreprises et n'est pas libre au téléchargement. De plus, aucune information n'est donnée quant à son prix. Enfin, bien que cette solution soit très certainement efficace, l'infrastructure nécessaire à sa mise en place est lourde et peu adaptée à mon projet en raison de l'intégration relativement difficile de cette infrastructure avec le projet et son manque de flexibilité concernant la gestion des demandes d'authentifications.



Figure 13: Exemple d'empreinte digitale utilisée lors d'une authentification biométrique

Par ailleurs, l'un des problèmes majeurs avec l'authentification par empreintes digitales est le manque de lecteurs compatibles avec les appareils mobiles, qu'ils fonctionnent avec Symbian OS ou Windows Mobile. En effet, il n'existe qu'une variété relativement limitée de lecteurs d'empreintes digitales, la plupart étant destinés à des stations Windows ou Linux. Les quelques lecteurs existants pour des appareils mobiles sont généralement directement intégrés dans l'appareil mobile, bien que quelques lecteurs externes existent, mais ceux-ci sont alors câblés en USB. Malheureusement, tous ces lecteurs possèdent des pilotes propriétaires, et l'utilisation de tels lecteurs pour ce projet nécessiterait d'écrire à nouveau leur pilote afin de pouvoir les utiliser depuis Java, tâche qui peut s'avérer très complexe et qui sortirait du cadre de ce projet.

11.4.1 Solution retenue

Bien qu'une authentification biométrique soit probablement le type d'authentification externe le plus efficace, le manque de matériel compatible ne m'a pas permis de réaliser cette fonctionnalité. Ainsi, le cahier des charges de ce projet ne comprend pas l'implémentation d'une méthode d'authentification externe, car aucune solution optimale n'a été trouvée. Le projet a toutefois été structuré de manière à pouvoir accueillir une méthode d'authentification externe très facilement ultérieurement.

Une solution a cependant été implémentée à titre d'exemple afin de démontrer le but de l'authentification externe. Cette solution se base sur la technologie Bluetooth permettant à l'appareil mobile de détecter une tentative de vol. Le principe est simple : en imaginant que l'utilisateur porte sur lui un second appareil Bluetooth, tel qu'un badge ou un téléphone mobile, l'appareil mobile se connecte

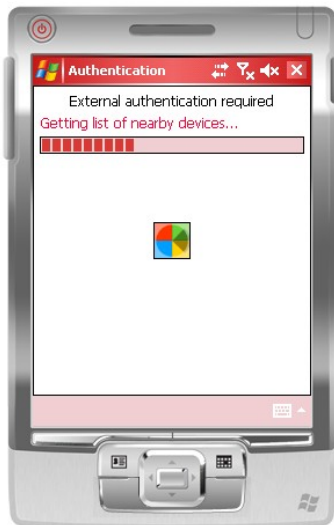


Figure 14: Authentification externe à l'aide de la technologie Bluetooth

à ce second appareil. En cas de vol de l'appareil mobile, la liaison entre l'appareil mobile et le badge se rompt en raison de la portée limitée de la technologie Bluetooth, signifiant que l'appareil mobile a été volé. Si ce cas se produit, alors l'appareil mobile bloque automatiquement tout accès au service.

En pratique, l'appareil mobile effectue périodiquement un balayage de tous les appareils Bluetooth aux alentours et enregistre l'adresse MAC (*Medium Access Control*) de ces appareils. Ce balayage est effectué à l'aide d'une librairie externe écrite en C++ et de JNI, car le Bluetooth est une fonctionnalité non implémentée dans la machine virtuelle Java IBM J9. Une fois le balayage effectué, une liste contenant l'adresse MAC des appareils détectés est envoyée au serveur. Le serveur contrôle alors si l'adresse MAC du badge (ou du téléphone) Bluetooth du médecin est contenue dans cette liste d'adresses. Si l'adresse est contenue dans la liste, l'authentification est valide. Dans le cas contraire, l'authentification est considérée comme invalide. Le client lourd tente alors de nouveaux balayages jusqu'à ce que le compte de l'utilisateur soit bloqué par le serveur.

11.5 Résumé des types d'authentification implémentés

- ◇ Authentification formelle : cette authentification est effectuée par la saisie du nom de l'utilisateur et de son mot de passe;
- ◇ Authentification matérielle : cette authentification (ou plus précisément identification) récupère le numéro IMEI du PDA Phone qui permet de l'identifier de manière unique. Cette identification est difficilement falsifiable;
- ◇ Authentification externe : la réalisation de ce projet n'inclut pas l'implémentation d'une authentification externe, en raison notamment du manque de matériel compatible pour créer une authentification biométrique. L'architecture du système est en revanche construite de manière à pouvoir ajouter très facilement une méthode d'authentification externe ultérieurement. Une authentification externe fonctionnant avec la technologie Bluetooth a toutefois été implémentée à titre d'exemple afin de démontrer une utilisation possible d'une telle authentification.

12. Système de mise à jour

La réalisation de ce projet comprend un système de mise à jour permettant d'ajouter des applications ou de mettre à niveau les applications déjà existantes sur le client mobile.

L'objectif du système de mise à jour est d'assurer que les fichiers installés sur le client lourd soient à jour et qu'ils soient valides au niveau de leur intégrité. Une application qui ne satisfait pas à ces deux critères peut présenter des failles de sécurité et l'utilisation d'une telle application peut compromettre la confidentialité des données et du service dans son ensemble.

12.1 Définition d'une application

Afin de mieux comprendre comment le système de mise à jour fonctionne, il est tout d'abord nécessaire de connaître la structure d'une application. Une application, telle que définie pour ce service, représente un ensemble de fichiers liés dont certains sont exécutables. Par exemple, si un fichier exécutable *Application1.jar* nécessite une librairie externe *commonDll.dll* pour s'exécuter, alors la création d'une application *Pack 1* contenant ces deux fichiers permet de garantir que les deux fichiers soient mis à jour et installés ensemble correctement. Un fichier peut par ailleurs être utilisé par plusieurs applications simultanément, comme c'est le cas du fichier *commonDll.dll* qui est utilisé à la fois par l'application *Pack 2* et l'application *Pack 3* dans le schéma 15.

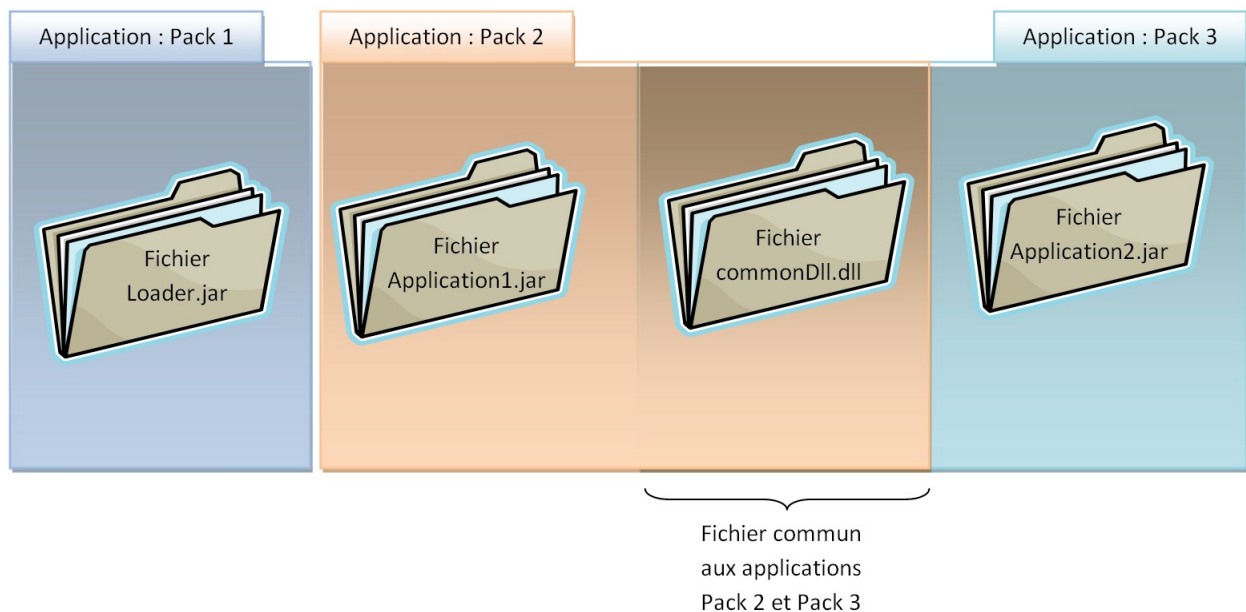


Figure 15: Exemple d'applications pour le service de mise à jour

Une application n'est ainsi pas exécutable en elle-même, mais ce sont les fichiers que l'application regroupe qui sont exécutables.

Remarque :

La notion « d'application » au sens « regroupement de fichiers » n'est valable que dans ce chapitre d'explications du service de mise à jour. Dans le but de faciliter la compréhension, le terme « application » dans les autres chapitres de ce document représente le fichier exécutable contenu dans l'application.

12.2 Types de mises à jour

Le système de mise à jour doit prendre en compte la contrainte principale liée aux appareils mobiles qui est le taux de transfert des données limité. En raison de cette contrainte, il est nécessaire de ne mettre à jour ou d'installer automatiquement que les applications qui le nécessitent vraiment (mises à jour prioritaires ou applications prioritaires) et de laisser l'utilisateur choisir les mises à jour ou les installations non prioritaires à effectuer. Nous pouvons alors définir trois catégories d'applications différentes :

- ◇ Catégorie 1 :
 - Applications prioritaires : une telle application est installée et mise à jour obligatoirement, sans prendre en compte l'avis de l'utilisateur. L'intégrité de ce type d'applications est contrôlée et assurée à chaque démarrage du client lourd.
- ◇ Catégorie 2 :
 - Applications non prioritaires dont la mise à jour est obligatoire : une telle application est mise à jour obligatoirement si une précédente version est présente sur l'appareil mobile. L'installation de ce type d'applications n'est cependant pas obligatoire et est laissée au choix de l'utilisateur. L'intégrité d'une telle application est contrôlée et assurée à chaque démarrage du client lourd si l'application est installée.
- ◇ Catégorie 3 :
 - Application non prioritaire (ou optionnelle) : ce type d'application peut ou non être installé ou mis à jour par l'utilisateur. L'intégrité d'une telle application ne peut pas être garantie.

12.3 Délai de transition

Un délai de transition peut être défini au niveau du serveur pour chaque groupe d'utilisateurs. Ce délai indique la durée après laquelle la mise à jour d'une application non prioritaire devient obligatoire. En d'autres mots, ce délai indique à partir de quel moment une application de la catégorie 3 devient une application de la catégorie 2.

Ce délai permet à un médecin en déplacement d'avoir la possibilité de ne pas mettre à jour une application dont la mise à jour vient d'être publiée, dans le cas par exemple où le médecin se trouverait dans un réseau à faible débit. Comme le fait de ne pas mettre à jour une application présente des risques de sécurité, le délai de transition permet de rendre une mise à jour obligatoire passé ce délai.

De manière plus précise, ce délai est comparé à la date de modification ou d'ajout de chaque fichier sur le serveur. Si une application non prioritaire possède au moins un fichier dont la mise à jour est obligatoire, alors la mise à jour de tous les fichiers de cette application devient obligatoire.

Par défaut, chaque fichier est soumis au délai de transition appliqué au groupe auquel appartient l'utilisateur. L'administrateur peut en revanche, s'il le souhaite, rendre la mise à jour d'un fichier obligatoire dès sa publication sur le serveur, ignorant ainsi le délai de transition¹⁰. Cela peut être utile dans le cas d'une mise à jour destinée à la correction d'une faille critique. L'administrateur peut aussi choisir de désactiver totalement ce délai de transition pour un ou plusieurs groupes d'utilisateurs s'il souhaite un maximum de sécurité. Dans un tel cas, l'intégrité de toutes les applications est assurée, car l'utilisateur n'a plus la possibilité de ne pas mettre à jour les applications¹¹.

¹⁰ Pour plus d'informations concernant l'ajout de fichiers et d'applications sur le serveur, voir le chapitre 19.3.10, page 105.

¹¹ Pour désactiver le délai de transition, se référer au chapitre 19.3.5, page 102

12.4 Contrôle des versions

Le contrôle de la version des fichiers au niveau du client s'effectue de la manière suivante :

- ◇ Le client lourd récupère la liste de toutes les applications que l'utilisateur est autorisé à utiliser. Pour chaque application, le client lourd récupère la liste des fichiers qui composent l'application, ainsi que le code haché de chaque fichier.
- ◇ Le client lourd calcule le code haché de chaque fichier installé à l'aide de l'algorithme de hachage MD5, puis compare le résultat de ce calcul avec la valeur de référence fournie par le serveur.
 - Si les codes hachés sont identiques, la version du fichier est à jour et est valide. Aucune mise à jour n'est nécessaire.
 - Si les codes hachés sont différents, le fichier en question est soit obsolète, soit corrompu. La décision de mettre à jour ou non ce fichier dépend ensuite de la catégorie à laquelle l'application contenant ce fichier appartient.

L'avantage d'utiliser le calcul du code haché pour contrôler la version du fichier est que cette méthode peut s'appliquer à tous les types de fichiers.

Aucun fichier de configuration contenant la liste des applications installées n'est stocké sur le client. Cela permet d'accroître la sécurité, car un tel fichier de configuration pourrait facilement être modifié dans le but de faire passer un fichier exécutable corrompu pour un fichier valide.

Remarque :

Le serveur ne possédant que le code haché de la version la plus récente des fichiers dans sa base de données, la détection d'un fichier invalide par le client ne permet pas de savoir si ce fichier est simplement une ancienne version ou s'il s'agit d'une nouvelle version qui a été corrompue. Cet inconvénient ne peut devenir un problème que lorsque le fichier en question appartient à une application de catégorie 3 (dont la mise à jour est optionnelle), auquel cas l'utilisateur pourrait choisir de ne pas mettre à jour l'application et donc d'utiliser une version potentiellement corrompue du fichier. Cela explique la raison pour laquelle l'intégrité d'une application de catégorie 3 ne peut pas être assurée. Le délai de transition a été implémenté afin de limiter la durée durant laquelle la mise à jour d'une application est optionnelle (chapitre 12.3). Ce problème est toutefois inexistant pour les applications de catégorie 1 ou 2, car les mises à jour de ces applications se font automatiquement, sans laisser le choix à l'utilisateur.

A titre de rappel, l'administrateur du service peut désactiver le délai de transition afin de rendre toutes les mises à jour obligatoires, ce qui garantit qu'aucune application ne possède de fichiers corrompus ou obsolètes, et donc qu'aucune application n'appartienne à la catégorie 3.

12.5 Types d'installation

Une application peut être définie comme n'étant pas installable automatiquement. Ceci pourrait être le cas de fichiers de taille très importante qui nécessiteraient trop de temps à être téléchargés par l'appareil mobile. Si une application n'est pas installable automatiquement et qu'elle doit être installée ou mise à jour, les cas de figure suivants peuvent se produire :

- ◇ L'application est prioritaire (catégorie 1) : une erreur s'affiche indiquant à l'utilisateur qu'il doit faire mettre à jour l'application par l'administrateur du service, puis le client se ferme automatiquement afin d'éviter tout risque de sécurité.

- ◇ L'application est non prioritaire, mais doit être mise à jour obligatoirement (catégorie 2) : un message d'erreur s'affiche indiquant à l'utilisateur qu'une application non installable automatiquement doit être mise à jour manuellement par l'administrateur, puis le client supprime l'application concernée afin que l'utilisateur puisse continuer à utiliser le client lourd sans risque de failles de sécurité.
- ◇ L'application est non prioritaire (catégorie 3) : un message d'avertissement s'affiche indiquant à l'utilisateur qu'une application nécessite d'être mise à jour manuellement par l'administrateur. L'application n'est cependant pas supprimée et peut continuer à être utilisée.

12.6 Contraintes de mise à jour

Le système de mise à jour doit garantir que chaque application installée possède tous les fichiers nécessaires à son exécution. Dans le cas contraire, l'application ne doit pas être disponible à l'exécution.

En sachant que certains fichiers peuvent être communs à plusieurs applications, les contraintes suivantes doivent être définies et respectées :

1. Si la mise à jour d'une application entraîne la modification d'un fichier commun entre cette application et une ou plusieurs autres applications, alors les autres applications doivent également être mises à jour.
2. En cas de suppression d'un fichier commun à plusieurs applications, toutes les applications utilisant ce même fichier doivent aussi être supprimées.

La première contrainte permet de garantir qu'aucune application n'ait qu'une partie de ses fichiers qui soient mis à jour, et donc un risque d'incompatibilité entre les nouveaux fichiers et les anciens fichiers. Prenons un exemple :

Un client mobile possède deux applications installées :

- ◇ Application A contenant les fichiers :
 - Application1.jar
 - commonDll.dll
- ◇ Application B contenant les fichiers :
 - Application2.jar
 - commonDll.dll

Imaginons que ces deux applications ne soient pas prioritaires, qu'aucun fichier ne soit à jour et que la mise à jour de ces fichiers ne soit pas obligatoire. Le client lourd proposerait à l'utilisateur de mettre à jour l'application A et B. Si l'utilisateur décidait ne mettre à jour que l'application A par exemple, alors le service de mise à jour téléchargerait les fichiers commonDll.dll et Application1.jar. Rien ne garantirait en revanche que l'ancien fichier Application2.jar soit compatible avec la nouvelle version du fichier commonDll.dll. Ainsi, même si l'utilisateur n'avait pas sélectionné la mise à jour de l'application B, cette dernière serait tout de même effectuée s'il existe une mise à jour pour le fichier Application2.jar, afin d'éviter tout problème lors de l'exécution de ce fichier.

La seconde contrainte permet de garantir qu'aucun fichier ne soit manquant à une application installée. Si un fichier venait à manquer, alors les applications utilisant ce fichier seraient supprimées afin d'éviter tout problème.

En reprenant l'exemple des applications ci-dessus, imaginons que les applications soient installées et que le fichier commonDll.dll soit supprimé manuellement par l'utilisateur lorsque le client lourd est

arrêté, depuis l'explorateur de fichier de Windows Mobile par exemple. Après le démarrage du client lourd, ce dernier proposerait à l'utilisateur d'installer les applications A et B, car le fichier commonDll.dll est manquant. Si l'utilisateur décide de n'effectuer aucune installation, alors les fichiers Application1.jar et Application2.jar risquent de ne plus fonctionner correctement.

Pour pallier ce problème, le service de mise à jour se termine par une vérification globale de toutes les applications installées. Ce processus de nettoyage supprime tous les fichiers des applications marquées comme étant à supprimer ainsi que tous les fichiers inutilisés. Les détails concernant ce nettoyage sont donnés dans le chapitre 12.8.

Dans le cas présent, les fichiers *Application1.jar* et *Application2.jar* seraient aussi supprimés.

12.7 Annulation des téléchargements

L'utilisateur a le choix d'annuler le téléchargement de fichiers appartenant à des applications non prioritaires, qu'elles doivent être mises à jour obligatoirement ou non (applications de catégorie 2 ou 3). Dans le cas où l'utilisateur décide d'annuler le transfert du fichier en cours de téléchargement, le fichier concerné est supprimé et l'application contenant ce dernier est marquée comme étant à supprimer à la fin du processus de mise à jour.

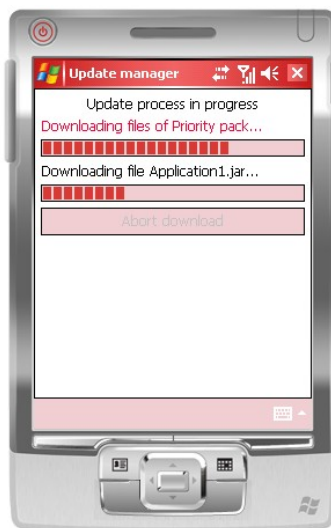


Figure 16: Mise à jour d'une application prioritaire

12.8 Nettoyage des fichiers

Le nettoyage des fichiers s'effectue en fin de processus de mise à jour. Le client lourd contrôle toutes les applications et tous les fichiers qui sont installés et les compare à la liste des applications et des fichiers que l'utilisateur est autorisé à utiliser provenant du serveur. Le client supprime ensuite tous les fichiers qui ne sont pas autorisés à être utilisés ainsi que tous les fichiers appartenant aux applications qui ont été marquées comme étant à supprimer lors des mises à jour (après une annulation de téléchargement par exemple). Il supprime également toutes les applications dont il manque au moins un fichier. Il est à noter qu'un fichier partagé entre plusieurs applications n'est supprimé que si aucune autre application légitime ne l'utilise.

Si l'un des fichiers ne peut pas être supprimé, par exemple en raison d'une protection d'écriture sur ce fichier, alors ce fichier est marqué comme étant à supprimer par la machine virtuelle à la fermeture

(normale) de celle-ci. Ce fichier, bien que toujours présent dans le répertoire du client lourd, n'est pas inclus dans la liste finale des fichiers qui sont autorisés à être exécutés et qui est transmise au chargeur d'applications. Cela permet de s'assurer que l'exécution de ce fichier ne soit pas possible.

12.9 Chargeur d'applications

Le chargeur d'applications permet de charger les applications installées sur le client lourd. La notion « d'applications » au sens regroupements de fichiers n'est utilisée que dans le cadre des mises à jour afin d'assurer que les fichiers soient téléchargés correctement, mais n'est plus utilisée pour la suite de l'exécution du client lourd. Le chargeur d'applications (implémenté dans le cadre d'un autre travail de diplôme) travaille au niveau des fichiers directement, c'est la raison pour laquelle il reçoit la liste des fichiers installés et autorisés à être exécutés lorsqu'il est instancié.

Remarque :

Pour des raisons de sécurité, le chargeur d'applications ne doit utiliser que la liste des fichiers autorisés à être exécutés transmise en paramètre lors de la création du chargeur. Il se peut en effet que certains fichiers non autorisés se trouvent encore dans le dossier du client lourd s'ils n'ont pas pu être supprimés, en raison par exemple d'une protection d'écriture.

12.10 Gestion des mises à jour

Le processus de mise à jour est représenté dans le schéma 18 « Diagramme d'activités des mises à jour et de l'installation des applications », à la page 49. Les étapes principales sont détaillées ci-dessous :

- ◇ Démarrage du client lourd : l'utilisateur démarre le client lourd.
- ◇ Contrôle de version de la machine virtuelle : le service de mise à jour vérifie si la version de la machine virtuelle est à jour. Dans le cas où le modèle ou la version de la machine virtuelle ne sont pas identiques à ceux stockés sur le serveur, alors le client lourd affiche un message d'erreur indiquant à l'utilisateur que la machine virtuelle doit être mise à jour par l'administrateur. Le client est ensuite fermé automatiquement. La raison de ce choix est due à la taille importante d'une machine virtuelle qui la rend difficilement transférable sur un appareil mobile. Ainsi, la mise à jour de la machine virtuelle se fera par une personne agréée, telle que l'administrateur du service.
- ◇ Contrôle de la version du client lourd : le client lourd se contrôle ensuite lui-même, en comparant son code haché à celui stocké sur le serveur. Si les codes hachés ne sont pas identiques, le client télécharge automatiquement la nouvelle version sur le serveur, puis l'installe en remplaçant l'ancienne version. Un redémarrage manuel est ensuite requis de la part de l'utilisateur afin de charger la nouvelle version en mémoire.
- ◇ Contrôle de la librairie externe : le même processus s'effectue pour la librairie externe nécessaire à l'authentification matérielle et externe. Aucun redémarrage n'est cependant requis pour ce fichier.
- ◇ Authentification : le client lourd effectue une authentification matérielle de l'appareil suivie d'une authentification formelle de l'utilisateur. Cette authentification permet, en plus de sécuriser l'accès au service, de connaître les droits de l'utilisateur et donc de savoir quelles sont les applications qu'il a le droit d'utiliser. Cette authentification n'est faite qu'après le contrôle de la version du client lourd et de sa librairie afin de s'assurer que ces fichiers soient à jour et que les éventuelles failles de sécurité soient corrigées avant l'authentification.

- ◇ Contrôle des applications et des fichiers : le client lourd récupère la liste de tous les fichiers qui sont installés dans son répertoire. Il récupère par ailleurs sur le serveur la liste de toutes les applications que l'utilisateur est autorisé à utiliser, ainsi que la liste des fichiers qui composent chaque application.
- ◇ Contrôle d'installation et de mise à jour des applications prioritaires : le client vérifie ensuite si des applications prioritaires doivent être installées ou mises à jour. Si tel est le cas, alors le client effectue les téléchargements et les installations nécessaires sans demander l'avis à l'utilisateur. Dans le cas où une application ne peut pas être installée automatiquement, le client affiche un message d'erreur et se ferme.
- ◇ Contrôle d'installation et de mise à jour des applications non prioritaires : le client vérifie si des applications de catégorie 2 (applications optionnelles) ou 3 (applications non prioritaires dont la mise à jour est obligatoire) peuvent être installées ou mises à jour. Si tel est le cas, une fenêtre de choix des applications s'ouvre afin que l'utilisateur puisse sélectionner les applications à installer/mettre à jour. Les applications de catégorie 3 sont cochées automatiquement si une mise à jour est requise et ne peuvent pas être décochées, car leur mise à jour est obligatoire. Une fois validée, la fenêtre de sélection se ferme et les applications sélectionnées sont téléchargées et installées.

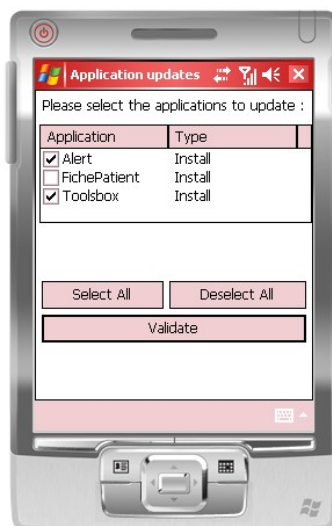


Figure 17: Fenêtre de sélection des mises à jour

- ◇ Nettoyage des fichiers : le client lourd supprime tous les fichiers des applications non autorisées à être exécutées ainsi que tous les fichiers des applications pour lesquelles il manque au moins un fichier.
- ◇ Démarrage du service de gestion des authentifications : le client lourd démarre le service de gestion des demandes d'authentifications et de contrôle des applications.
- ◇ Démarrage du chargeur d'application : le client lourd démarre le chargeur d'applications en lui donnant en paramètre la liste des fichiers installés et autorisés à être exécutés. Le processus de mise à jour est alors terminé.

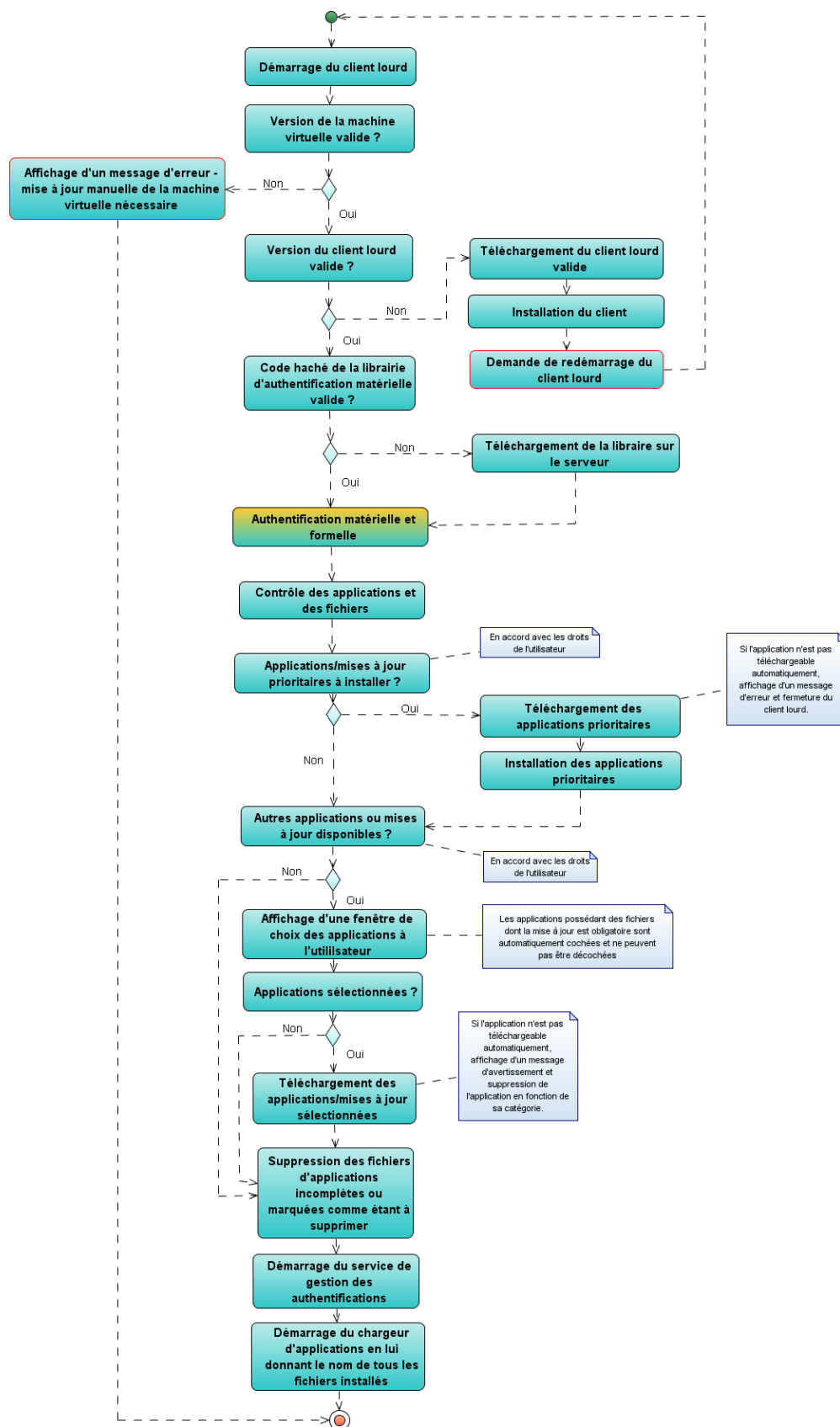


Figure 18: Diagramme d'activités des mises à jour et de l'installation des applications

13. Communications entre le client lourd et le serveur

13.1 Format des messages

Au niveau des échanges des messages entre le client et le serveur, il existe deux solutions principales : les *sockets* ou les services Web, avec *XML* par exemple. Les *sockets* offrent une bonne interaction entre le client et le serveur, car il est possible pour le serveur d'effectuer des requêtes vers le client si une *socket* a été établie dans les deux sens. Bien qu'une *servlet* avec *XML* ne permette pas une telle interaction, l'infrastructure pour gérer les connexions est moins lourde que pour les *sockets*, car les *servlets* fonctionnent sur la base d'un serveur Web qui gère lui-même les connexions. Comme il n'est pas nécessaire que le serveur initie des connexions vers le client dans le cadre de cette partie du projet, la solution des *servlets* avec *XML* est retenue.

Du côté du serveur, *JAXB* (Java Architecture for XML Binding) est une solution de *SUN* qui permet d'effectuer la conversion entre les classes Java et un document *XML* très aisément et automatiquement. Cette technologie requiert qu'un schéma de type « *XML Schema* » soit créé manuellement afin que le parseur *JAXB* puisse créer une classe Java à partir du schéma. La conversion entre les objets Java et le document *XML* est ensuite effectuée automatiquement par *JAXB*. Cette technologie n'est malheureusement pas disponible pour *Java ME*, c'est pourquoi il faut utiliser un parseur *XML* standard du côté du client. Un tel parseur n'est pas non plus intégré dans *Java ME*, mais il existe plusieurs parseurs téléchargeables et prévus à cet effet [20] :

Name	License	Size	MIDP	Type
ASXMLP 020308	Modified BSD	6 kB	yes	push, model
kXML 2.0 alpha	EPL	9 kB	yes	pull
kXML 1.2	EPL	16 kB	yes	pull
MinXML 1.7	BSD	14 kB	no	push
NanoXML 1.6.4	zlib/libpng	10 kB	patch	model
TinyXML 0.7	GPL	12 kB	no	model
Xparse-J 1.1	GPL	6 kB	yes	model

J'ai retenu le parseur *kXML* version 2 en raison de la documentation qui est disponible pour ce parseur et pour la licence qui n'est pas trop restrictive (*EPL* - Eclipse Public License), en opposition à la licence *GPL*. Cette licence requiert seulement que le code faisant partie de la licence *EPL* reste *EPL*, le reste du code (que nous avons ajouté nous-mêmes) peut être soumis à la licence de notre choix, donc sans restriction [21].

Pour effectuer les requêtes entre le client mobile et le serveur, un document *XML* est utilisé. Le schéma *xsd* des documents *XML* est le suivant :

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
  <xs:element name="message" type="messageContent"/>
  <xs:complexType name="messageContent">
    <xs:choice>
      <xs:element name="methodCall" type="typeMethod"/>
      <xs:element name="methodReturn" type="typeMethod"/>
    </xs:choice>
  </xs:complexType>
  <xs:complexType name="typeMethod">
    <xs:sequence>
```

```

<xs:element name="methodName" type="xs:string"/>
<xs:element name="parameter" type="typeParameter" minOccurs="0"
maxOccurs="unbounded"/>
</xs:sequence>
<xs:attribute name="id" type="xs:string" use="required"/>
</xs:complexType>
<xs:complexType name="typeParameter">
  <xs:sequence>
    <xs:element name="parameterName" type="xs:string"/>
    <xs:element name="type" type="xs:string"/>
    <xs:element name="value" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
</xs:schema>

```

L'équivalent graphique du schéma est le suivant :

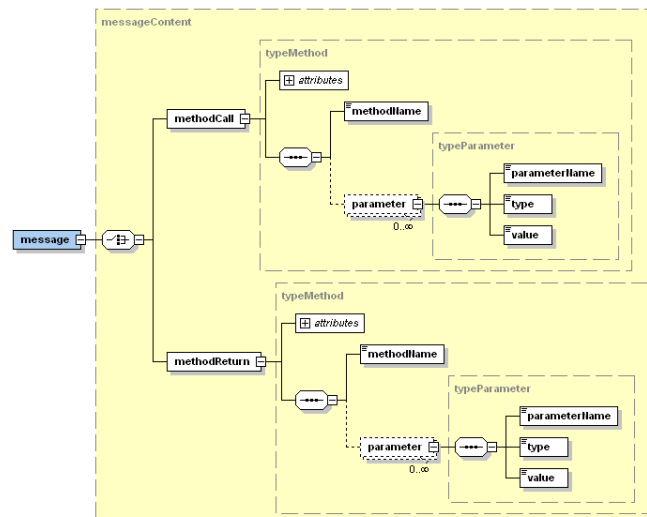


Figure 19: Représentation graphique du schéma XSD

Ce schéma définit précisément la syntaxe que doivent suivre les documents XML. Ceux-ci peuvent soit inclure un appel de méthode (pour le client mobile), soit un retour de méthode (pour le serveur). Ce choix se fait à l'aide de la balise `<xs:choice>`. De plus, chaque appel ou retour de méthode peut être composé de 0 à N paramètres qui ont un nom, un type et une valeur. Enfin, chaque appel ou retour de méthode possède un attribut « id » qui contient l'identifiant de sécurité du client lourd. Cet identifiant est généré par le serveur lors d'une authentification réussie du client et sert de contrôle pour le serveur de sécurité tout au long de la connexion¹².

Un document XML découlant de ce schéma permet ainsi d'effectuer un appel de méthode ou un retour de méthode ayant des paramètres de types simples, tels que des entiers, du texte ou encore des booléens, ce qui est bien adapté aux échanges de messages entre les clients et les serveurs dans ce projet.

Voici un exemple d'appel de méthode en XML:

¹² Les détails concernant le fonctionnement de l'identifiant de sécurité sont donnés au chapitre 13.3, page 54.

```
<?xml version="1.0" encoding="UTF-8"?>
<message xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation=".\\message.xsd">
  <methodCall id="secId713826217fb2a2cbdbfa0c7249f9c0d1">
    <methodName>checkPassword</methodName>
    <parameter>
      <parameterName>userLogin</parameterName>
      <type>String</type>
      <value>dupont</value>
    </parameter>
    <parameter>
      <parameterName>userPassword</parameterName>
      <type>String</type>
      <value>123456</value>
    </parameter>
  </methodCall>
</message>
```

La balise `methodCall` définit un nouvel appel. Dans cet exemple, l'appareil mobile souhaite appeler la méthode «`checkPassword`» en lui donnant le paramètre «`userLogin`» de type «`String`» et valant «`dupont`» et un autre paramètre «`password`» de type «`String`» et valant «`123456`».

Voici une réponse possible du serveur :

```
<?xml version="1.0" encoding="UTF-8"?>
<message xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation=".\\message.xsd">
  <methodReturn id="secId713826217fb2a2cbdbfa0c7249f9c0d1">
    <methodName>checkPassword</methodName>
    <parameter>
      <parameterName>userIsAuthenticated</parameterName>
      <type>boolean</type>
      <value>true</value>
    </parameter>
  </methodReturn>
</message>
```

Ici, l'exemple de réponse retourne un paramètre «`userIsAuthenticated`» de type «`boolean`» valant «`true`», ce qui signifie que l'utilisateur est authentifié.

13.2 Sécurisation des communications

Afin de sécuriser les échanges des données entre les appareils mobiles et le serveur de sécurité, l'implémentation d'un protocole de chiffrement des données est nécessaire. Le protocole TLS (Transport Layer Security), connu aussi sous l'ancien nom SSL (Secure Socket Layer), est une technologie de chiffrement des données très répandue et efficace actuellement.

13.2.1 Rappel du fonctionnement du protocole TLS

La technologie TLS repose sur un chiffrement asymétrique et sur un chiffrement symétrique des données. Le chiffrement asymétrique utilise une clé publique (disponible à tous) et une clé privée

(confidentielle), la clé publique permettant de déchiffrer les données chiffrées avec la clé privée, et vice versa. Quant au chiffrement symétrique, il utilise une paire de clés secrètes qui permettent à la fois de chiffrer et de déchiffrer les données.

Les clés asymétriques sont généralement utilisées pour établir un canal de communication sécurisé sur lequel des clés symétriques peuvent être négociées et générées. Les clés symétriques sont alors utilisées pour échanger de gros volumes de données, en raison de la moins grande complexité de l'algorithme de chiffrement symétrique par rapport à l'algorithme de chiffrement asymétrique.

De manière plus détaillée, les étapes suivantes sont effectuées lors de la connexion du client vers le serveur [22]:

1. Le client se connecte sur le serveur utilisant TLS. Le client fournit au serveur la liste des algorithmes de chiffrement et de hachage dont il dispose.
2. Le serveur sélectionne l'algorithme de chiffrement et de hachage le plus sûr qu'il possède en commun avec le client, puis envoie au client sa clé publique signée par une autorité de certification (*Certificate Authority*, CA).
3. Le client peut vérifier la validité du certificat s'il le souhaite auprès de la CA, puis il génère une paire de clés symétriques qu'il enverra au serveur, en les chiffrant préalablement à l'aide de la clé publique du serveur.
4. Le serveur reçoit puis déchiffre les clés symétriques à l'aide de sa clé privée. Le client et le serveur possèdent alors une paire de clés symétriques communes qu'ils utiliseront pour effectuer les échanges chiffrés des données.

13.2.2 Création des certificats

La marche à suivre pour créer les certificats du serveur de sécurité, du serveur de fichiers et du client lourd est donnée au chapitre 19.6, page 113.

13.2.3 Remarque concernant l'implémentation de TLS

Tout au long du projet, il s'est avéré que la transmission de tableaux d'octets générait parfois une erreur du type « *invalid padding* » au niveau SSL. Après de longues recherches et essais, il s'est avéré que cette erreur était due à la longueur des données envoyées. En effet, en fonction des données en octets à envoyer, il se peut que le protocole TLS requière un remplissage (*padding*) des données manquantes afin de pouvoir chiffrer les données. La gestion de ce remplissage semble cependant ne pas fonctionner correctement sur la machine virtuelle utilisée au niveau du client. Cela peut être dû à une gestion différente du *padding* au niveau du client qu'au niveau du serveur, mais il n'a malheureusement pas été possible de trouver plus d'informations à ce sujet. Cette erreur ne survient que lors de l'envoi de données du serveur vers le client. L'envoi des données du client vers le serveur ne pose quant à lui aucun problème.

Afin de résoudre ce problème, il faut éviter d'entrer dans la situation où un remplissage est nécessaire. Pour ce faire, il faut que la taille des données à envoyer soit égale à un multiple de 8 octets. Un algorithme simple a été mis en place afin de trouver la taille du paquet multiple de 8 octets maximale qui peut être envoyée, puis l'envoi des données restantes s'effectue octet par octet. L'algorithme fonctionne de la manière suivante, en prenant comme exemple un fichier de 3'765 octets à envoyer :

- ◇ Calcul de la taille de paquet maximale de 8 octets qu'il est possible d'envoyer par rapport à la taille des données et envoi de celui-ci :
 - Nombre de fragments = $3'765 / 8 = 470$ (division entière)

- Multiplications des fragments par 8 : $470 * 8 = 3'760$ octets
- Envoi du bloc de 3'760 octets au client
- ◇ Calcul du nombre d'octets restants à envoyer et envoi de ceux-ci octet par octet :
 - Envoi des 5 octets restants octet par octet

Il faut noter que les données restantes à envoyer ne dépassent jamais 7 octets. Cette façon de transmettre permet ainsi de garder un bon niveau de performances et d'éviter l'erreur susmentionnée.

13.3 Identifiant de sécurité

13.3.1 But

L'identifiant de sécurité permet de restreindre l'accès aux méthodes au niveau du serveur, ainsi que de reconnaître les clients authentifiés de ceux qui ne le sont pas.

Sans autre moyen de protection, une servlet n'est pas restreinte d'accès. Ainsi, toute personne connaissant la syntaxe XML utilisée par le serveur pourrait envoyer des requêtes à celui-ci dans le but par exemple d'accéder à des données sensibles. Il est donc nécessaire de reconnaître les requêtes légitimes pour leur donner accès aux méthodes sécurisées et de bloquer les autres requêtes. Pour ce faire, lorsqu'un client s'authentifie correctement de manière formelle sur le serveur, ce dernier génère deux identifiants qu'il transmet ensuite au client :

- ◇ Le premier identifiant est un identifiant de sécurité pour le client lourd qui permet au client de sécurité de communiquer avec le serveur de sécurité. Le serveur utilise cet identifiant afin de reconnaître les clients valides de ceux qui ne le sont pas. Au niveau du serveur, l'identifiant réfère par ailleurs à un objet contenant le login de l'utilisateur et le numéro IMEI de son appareil mobile dans le but d'éviter au client de devoir envoyer ces informations à chaque requête. Cet identifiant reste confiné à l'intérieur du client de sécurité une fois généré et n'est divulgué à aucune application externe.
- ◇ Le second identifiant de sécurité est généré pour être utilisé par les applications externes. Lorsqu'une telle application externe désire initier une connexion vers son serveur, elle va au préalable demander au client de sécurité de lui fournir l'identifiant de sécurité pour les applications. Le client effectue alors un contrôle de validité de l'application en collaboration avec le serveur de sécurité, et si l'application est valide (si l'utilisateur est autorisé à l'utiliser), le client de sécurité lui retourne l'identifiant de sécurité. Cet identifiant est ensuite inclus dans la requête de l'application qui est transmise au serveur de l'application. Ce dernier peut ensuite envoyer une requête au serveur de sécurité afin de savoir si l'identifiant est valide ou s'il faut rejeter la requête.

Ainsi, le premier identifiant n'est utilisé que par le client lourd alors que le second identifiant n'est utilisé que par les applications externes au client lourd. L'avantage d'avoir deux identifiants distincts est d'éviter qu'une faille de sécurité au niveau d'une application externe ne puisse être utilisée dans le but de récupérer l'identifiant de sécurité du client lourd, auquel cas des requêtes illégitimes sur le serveur de sécurité pourraient être effectuées. Une telle faille permettrait par exemple de déconnecter un utilisateur du serveur de sécurité à son insu et donc d'effectuer un déni de service.

Deux contraintes doivent cependant être respectées pour ces deux identifiants : d'une part, ils doivent rester confidentiels entre le client et le(s) serveur(s). Cela ne pose pas de problème particulier en raison de l'utilisation d'une connexion sécurisée. D'autre part, l'identifiant doit être assez complexe pour qu'il ne soit pas possible de le reconstituer ou de le deviner facilement. La méthode utilisée pour générer cet

identifiant, expliquée ci-dessous, devrait permettre d'assurer une protection suffisamment efficace.

13.3.2 Génération des identifiants de sécurité

Les identifiants de sécurité sont générés à chaque fois que l'utilisateur effectue une authentification formelle. La raison de ce choix tient au fait que l'authentification formelle nécessite plusieurs paramètres : le nom de l'utilisateur, son mot de passe ainsi que le numéro IMEI de l'appareil mobile qu'il est en train d'utiliser. Toutes ces informations, en addition au temps actuel en millisecondes et à un nombre aléatoire, sont combinées afin de générer un identifiant unique qui est alors haché à l'aide de l'algorithme MD5. L'illustration de ce processus est donnée dans la figure 20 «Génération des identifiants de sécurité».

Lorsqu'une authentification formelle s'effectue alors qu'un identifiant existe déjà pour cet utilisateur, alors l'ancien identifiant est remplacé par le nouvel identifiant.

De plus, chaque identifiant possède un délai de péremption. Si ce délai est échu, l'identifiant est automatiquement supprimé lors du nettoyage périodique des sessions. Un utilisateur essayant d'accéder au serveur en ayant un identifiant de sécurité échu ou invalide se verra refusé d'accès. Il est à noter que le délai de validité des identifiants peut être paramétré par l'administrateur du service¹³.

13.3.3 Sécurité de l'identifiant

La génération de l'identifiant de sécurité est difficile à deviner, car elle combine plusieurs informations différentes dont certaines ne sont connues que par l'utilisateur, telles que son mot de passe. De plus, l'ajout de l'heure en millisecondes, l'ajout d'un nombre aléatoire et le hachage de toutes ces informations rendent la falsification d'un tel identifiant très difficile.

La solution qui vient alors à l'esprit est d'effectuer une attaque par force brute, ce qui signifie que la personne malintentionnée tente de trouver un identifiant valide en essayant toutes les combinaisons possibles. La formule permettant de calculer le nombre de combinaisons possibles d'un mot de passe est donnée par :

$$\text{nombre moyen de combinaisons} = \frac{c^l}{2}$$

où l est le nombre de lettres composant l'identifiant et c la taille de l'alphabet utilisé. La division par deux représente la moyenne du nombre de combinaisons à essayer.

Sachant que le code haché contient 32 caractères hexadécimaux, le nombre de combinaisons à trouver en moyenne est alors :

$$\frac{16^{32}}{2} = 1.701 \cdot 10^{38}$$

En imaginant pouvoir tester 100 identifiants à la seconde, ce qui correspond à envoyer 100 requêtes par secondes au serveur, il faudrait

$$\frac{16^{32}}{2 \cdot 100 \cdot 3600 \cdot 24 \cdot 365} = 53'951'415'354'030'070'944'852'645'775 \text{ années en moyenne avant de trouver}$$

la bonne combinaison par force brute.

Sachant que l'identifiant se renouvelle à chaque authentification formelle, soit toutes les quatre heures environ, il est possible d'affirmer que cette solution est suffisamment sûre contre une attaque par force brute.

¹³ Pour modifier ce délai, se référer au chapitre 19.3.11, page 107.

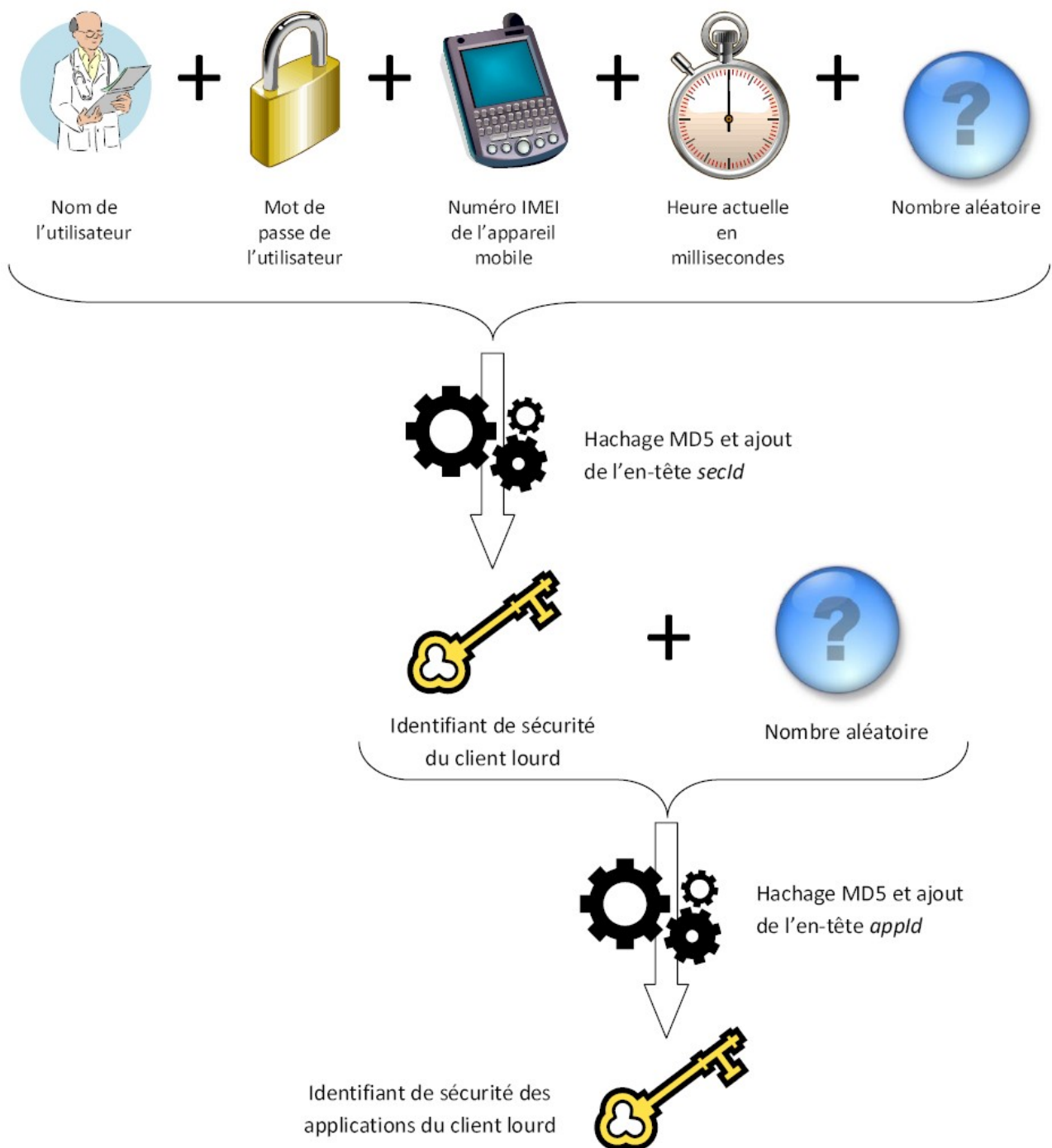


Figure 20: Génération des identifiants de sécurité

Lien vers l'image du point d'interrogation : [23]

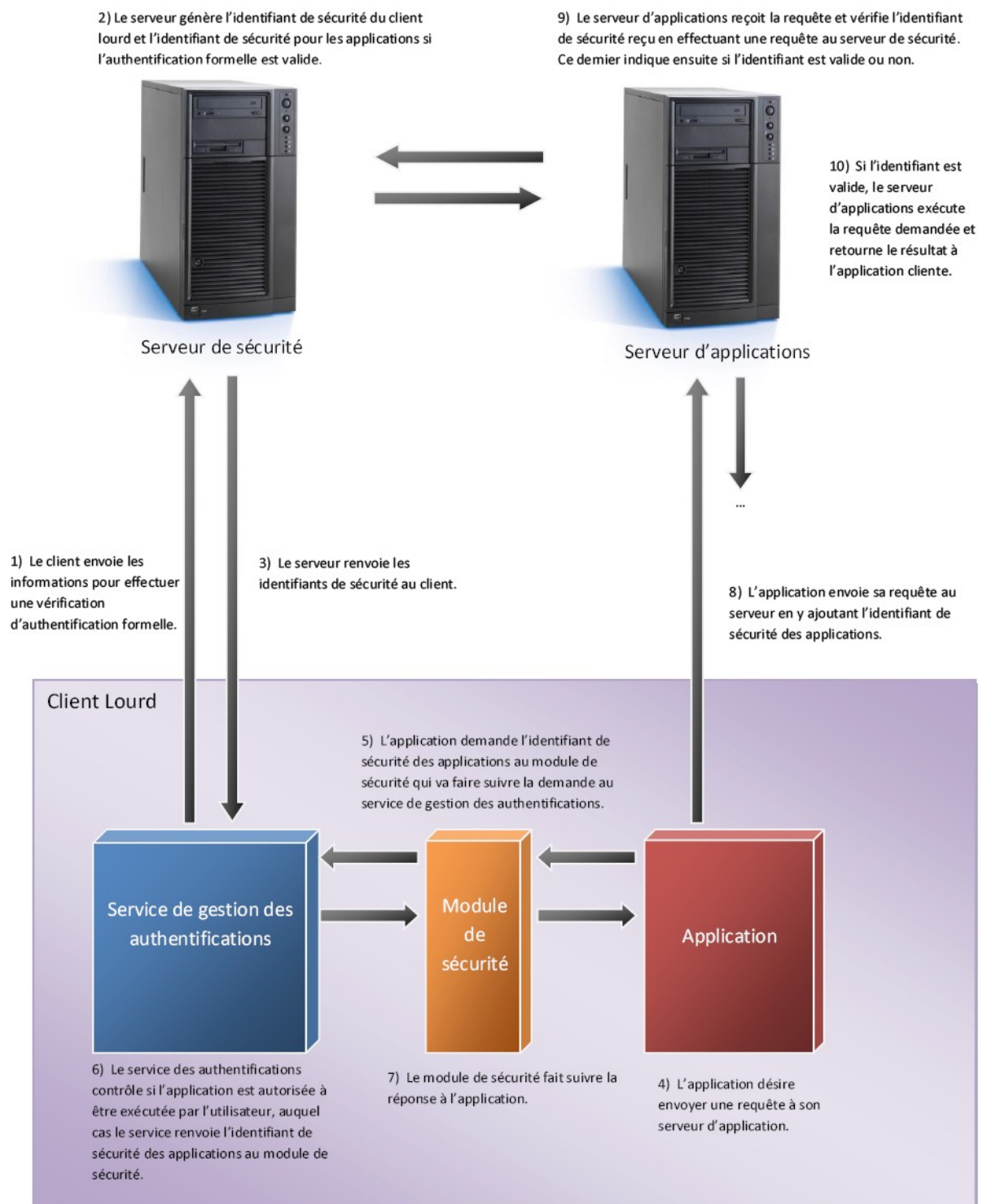


Figure 21: Création et échange des identifiants de sécurité

14. Service d'administration

Le service d'administration permet à l'administrateur de gérer l'ensemble du service de sécurité.

14.1 Architecture

La solution retenue pour la réalisation du service d'administration est la solution basée sur un site Internet. Une solution alternative aurait été d'implémenter une application riche en Java par exemple offrant les fonctionnalités de paramétrage du service de sécurité, mais il aurait alors aussi fallu gérer la problématique des mises à jour de cette application. Une telle application nécessite par ailleurs aussi qu'une machine virtuelle soit installée pour pouvoir être utilisée, ce qui peut être relativement contraignant en fonction de l'environnement dans lequel l'administrateur du service se trouve.

En revanche, un site Internet contenant les diverses possibilités de paramétrage du service de sécurité ne requiert qu'un navigateur Internet pour y accéder et n'est pas soumis aux problèmes d'anciennes versions. C'est la raison pour laquelle j'ai opté pour la solution basée sur un site Internet.

Au niveau des technologies utilisées, j'ai choisi les technologies des JSP et des servlets pour implémenter le service d'administration, car le serveur de sécurité est écrit en Java, ce qui simplifie les interactions entre les deux services.

14.2 Sécurité

L'accès au service d'administration requiert une authentification formelle de l'administrateur. Dans ce but, un champ a été prévu afin que l'administrateur puisse s'authentifier sur le site Internet lorsqu'il accède pour la première fois à celui-ci.

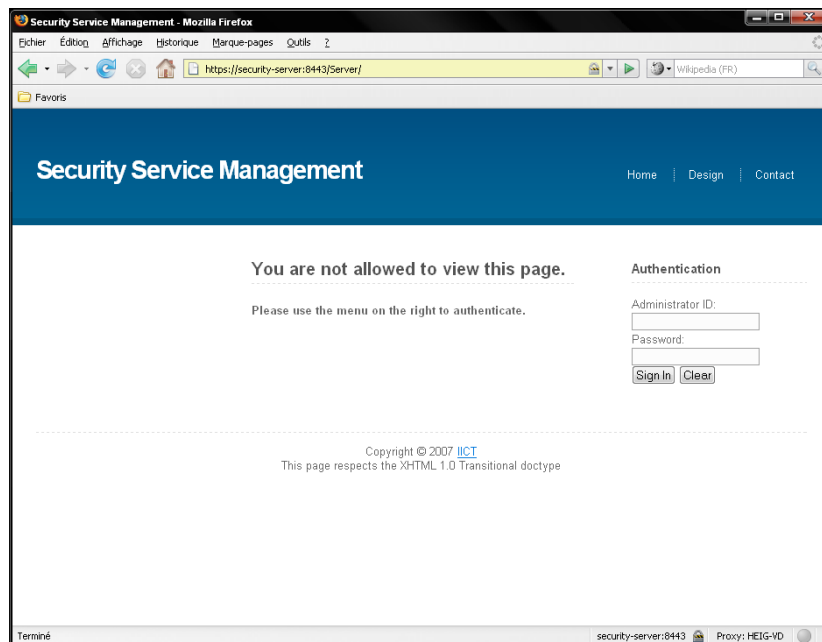


Figure 22: Fenêtre principale d'authentification

Du point de vue technique, afin de garantir qu'aucune page protégée ne puisse être atteinte sans authentification valide, le contrôle de la session de l'utilisateur et de sa bonne authentification a été inclus dans un fichier JSP d'en-tête et un fichier JSP de pied de page, les deux fichiers travaillant

ensemble. Toutes les pages JSP désirant ensuite être sécurisées n'ont qu'à inclure ces deux fichiers. Si la session de l'utilisateur n'est pas valide, alors l'en-tête remplace la page qui devrait s'afficher par un simple message d'erreur.

Cette méthode à l'avantage d'alléger et de faciliter le contrôle d'authentification, car toutes les autres pages JSP n'ont plus à se préoccuper de l'authentification du moment où elles incluent les deux fichiers susmentionnés.

14.3 Fonctionnalités

Le service d'administration permet à l'administrateur de gérer la totalité du serveur de sécurité.

La liste des fonctionnalités offertes par ce service est donnée ci-dessous :

- ◇ Gestion des utilisateurs autorisés à se connecter au service de sécurité;
- ◇ Gestion des groupes d'utilisateurs;
- ◇ Gestion des paramètres de sécurité régissant le comportement des authentifications et des mises à jour des clients lourds;
- ◇ Gestion des appareils mobiles autorisés à se connecter au service de sécurité;
- ◇ Gestion des attributions des appareils mobiles aux utilisateurs;
- ◇ Gestion des machines virtuelles autorisées à être utilisées dans le service de sécurité;
- ◇ Gestion des systèmes d'exploitation autorisés à être utilisés dans le service de sécurité;
- ◇ Gestion des applications et des fichiers qu'elles contiennent;
- ◇ Gestion des droits d'exécution des applications par les groupes d'utilisateurs;
- ◇ Gestion des paramètres de configuration généraux du serveur de sécurité;

Tous les détails concernant le fonctionnement de ces fonctionnalités sont donnés dans le manuel d'utilisation du service d'administration dans le chapitre 19.3 à la page 100.

14.4 Structuration

La figure 23 représente les différentes parties du site Web d'administration :

- ◇ La partie d'en-tête représentée en rouge dans la figure est affichée pour chaque page. Elle contient un lien permettant de retourner au menu principal, un second lien référant à la source d'où la mise en page du site a été inspirée et un lien de contact pour écrire un email en cas de problème.
- ◇ La seconde partie est le menu d'authentification représenté en jaune dans la figure. Ce menu gère la demande d'authentification lorsque l'administrateur n'est pas encore authentifié et affiche un lien permettant de se déconnecter si l'authentification est valide.
- ◇ Le menu des tâches à gauche, représenté en vert dans la figure, contient la liste de toutes les tâches que l'administrateur peut effectuer. Ces tâches sont regroupées par catégories afin de faciliter la navigation et sont accessibles depuis toutes les pages. Ce menu est caché lorsque l'administrateur n'est pas authentifié.
- ◇ La partie centrale, représentée en bleu dans la figure, est la page affichant la tâche sélectionnée par l'administrateur. Cette page est remplacée par un message d'erreur si l'administrateur tente d'y accéder sans authentification.

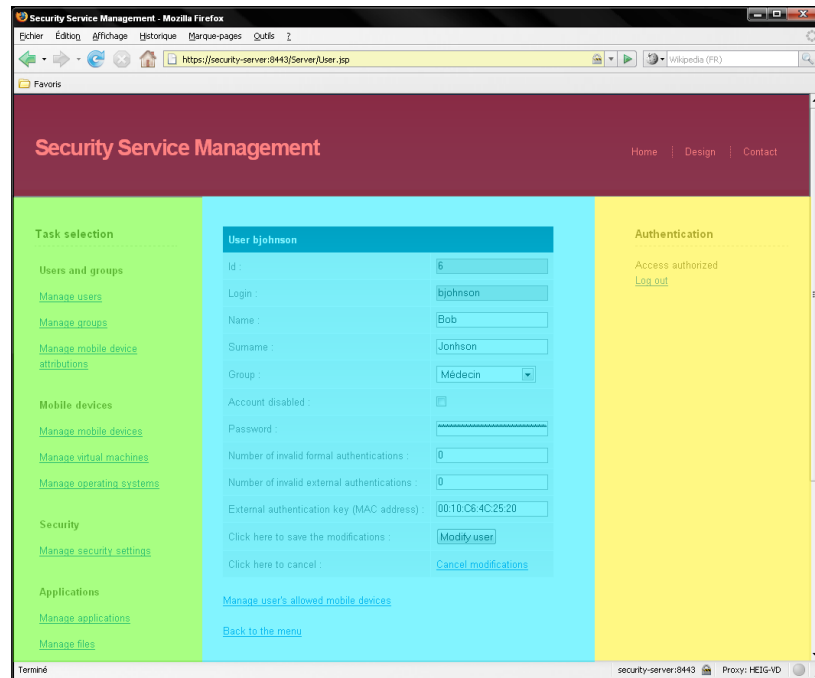


Figure 23: Structuration du site Internet d'administration

14.5 Mise en page

La mise en page et le design du site sont basés sur une feuille de style CSS (*Cascading Style Sheets*). Cette CSS provient d'un site Internet mettant à disposition des feuilles de style gratuitement [24]. La seule contrainte au niveau de la licence est qu'un lien vers le site source soit ajouté sur chaque site utilisant l'une de ces CSS.

L'avantage d'utiliser une feuille de style pour mettre en page le site est qu'une telle solution offre une bonne indépendance du contenu du site avec sa mise en page. Au niveau du code HTML, seules les classes définissant quel style appliquer à quel élément sont définies afin de mettre en page un élément, tel qu'une table ou un formulaire. Il serait ainsi relativement simple de remplacer la CSS dans le but de modifier la mise en page du site, tout en n'ayant que peu de code HTML à modifier.

14.6 Contrôles de saisie

Le contrôle des saisies dans les formulaires est effectué à l'aide de code JavaScript contenu dans les pages HTML envoyées au client. L'avantage de contrôler les champs des formulaires à l'aide de JavaScript est que ce traitement est effectué au niveau du client et non au niveau du serveur, offrant ainsi une meilleure réactivité et une diminution du trafic vers le serveur. Bien qu'il soit en principe possible de contourner JavaScript au niveau du client, et donc de contourner les contrôles des valeurs saisies, les champs critiques sont à nouveau contrôlés par le serveur, ce qui diminue les risques de saisies invalides. D'autre part, étant donné que seul l'administrateur du service a accès à ces formulaires, le risque que ce dernier contourne volontairement des systèmes de protection dans le but de nuire au service dont il est responsable est très faible.

Enfin, les champs critiques, tels que le champ d'authentification de l'administrateur, sont contrôlés afin d'éviter toute tentative d'injection SQL¹⁴.

14 L'injection SQL consiste à utiliser des caractères spéciaux dans un champ de saisie d'un site Internet dans le but d'insérer des commandes pouvant altérer les données contenues dans la base de données.

15. Base de données

15.1 Modèle conceptuel

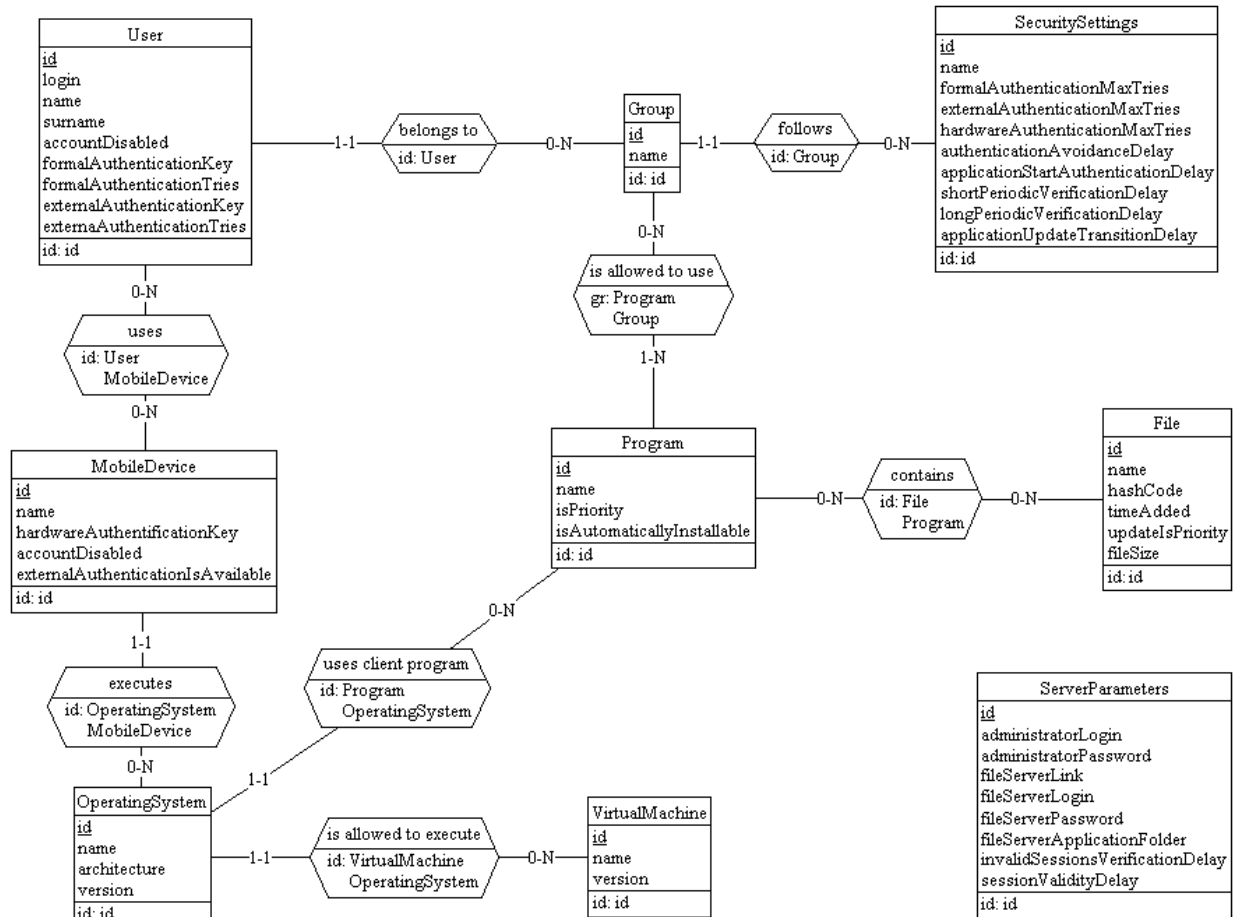


Figure 24: Schéma conceptuel de la base de données

La figure 24 représente le modèle conceptuel de la base de données qui a été implémentée dans le cadre de ce projet.

La base de données est composée des neuf tables suivantes :

- ◇ **Table User :** cette table contient tous les attributs concernant l'utilisateur, comme son login, son nom, son prénom, un paramètre définissant si le compte est activé ou désactivé (*accountDisabled*), le mot de passe de l'utilisateur (*formalAuthenticationKey*) ou encore son identifiant externe (*externalAuthenticationKey*). Cette table contient également le nombre de tentatives d'authentifications formelles échouées (*formalAuthenticationTries*) et externes (*externalAuthenticationTries*) effectuées par l'utilisateur. Au niveau des liens avec les autres tables, un utilisateur peut être autorisé à utiliser plusieurs appareils mobiles et il ne peut appartenir qu'à un seul groupe à la fois.
- ◇ **Table MobileDevice :** cette table comporte tous les éléments propres à l'appareil mobile, tels que le nom de l'appareil, son code IMEI (*hardwareAuthenticationKey*), un paramètre définissant si le compte est activé ou désactivé (*accountDisabled*) et une propriété permettant de déterminer si l'appareil mobile possède un module d'authentification externe ou non

(*externalAuthenticationIsAvailable*).

- ◇ Table Group : cette table permet de regrouper les utilisateurs en fonction des applications qu'ils utilisent et des droits qui leur sont accordés. Chaque groupe ne suit qu'un seul ensemble de règles (*SecuritySettings*) et peut être autorisé à exécuter 0 à N applications.
- ◇ Table Program : cette table représente une application telle que nous l'avons vue au chapitre 12.1. Elle contient toutes les informations relatives aux applications : nom de l'application, un champ permettant de savoir si cette application est prioritaire ou non, donc si elle est installée avec ou sans l'avis de l'utilisateur (*isPriority*) et un champ pour savoir si cette application peut s'installer automatiquement ou si elle doit être installée manuellement par l'administrateur (*isAutomaticallyInstallable*). Une application peut contenir 0 à N fichiers.
- ◇ Table File : en liaison avec la table Program, cette table représente un fichier contenu dans une application. Un tel fichier est défini par un nom, un code haché permettant de vérifier l'intégrité du fichier (*hashCode*), la date d'ajout du fichier sur le serveur (*timeAdded*), un paramètre permettant de définir si la mise à jour de ce fichier est obligatoire ou non (*updateIsPriority*) et la taille du fichier (*fileSize*).
- ◇ Table SecuritySettings : cette table contient tous les paramètres de sécurité appliqués au service dans son ensemble :
 - Un nom permettant de reconnaître l'ensemble de règles;
 - Un nombre qui définit le nombre maximum d'authentifications invalides formelles avant que le compte de l'utilisateur ne soit bloqué (*formalAuthenticationMaxTries*);
 - Un nombre qui définit le nombre maximum d'authentifications invalides externes avant que le compte de l'utilisateur ne soit bloqué (*externalAuthenticationMaxTries*);
 - Le délai d'évitement (*authenticationAvoidanceDelay*) qui définit le délai après lequel une nouvelle authentification formelle est demandée à la suite d'un évitement d'une telle authentification;
 - Le délai de démarrage des applications après lequel le démarrage d'une nouvelle application déclenche une demande d'authentification externe (*applicationStartAuthenticationDelay*);
 - Le délai d'authentification court périodique entre deux authentifications externes (*shortPeriodicVerificationDelay*);
 - Le délai d'authentification long périodique entre deux authentifications matérielles et formelles (*longPeriodicVerificationDelay*);
 - Le délai de transition des mises à jour après lequel une mise à jour optionnelle devient obligatoire (*applicationUpdateTransitionDelay*);
- ◇ Table OperatingSystem : toutes les informations concernant les systèmes d'exploitation utilisés par les appareils mobiles sont contenues dans cette table. Un système d'exploitation est ainsi défini par un nom, une version et une architecture (par exemple ARM ou MIPS). Un système d'exploitation n'est autorisé à utiliser qu'un seul client lourd et ne peut utiliser qu'une seule machine virtuelle. Il est à noter que le client lourd est aussi une application, c'est la raison pour laquelle la table *OperatingSystem* possède un lien vers la table *Program*.
- ◇ Table VirtualMachine : une machine virtuelle peut être définie par un nom et une version. Cette table permet de vérifier si l'utilisation d'une machine virtuelle donnée est autorisée dans le service de sécurité ou non. Elle permet par ailleurs de vérifier la version de la machine virtuelle afin de savoir si des mises à jour sont requises.

- ◇ Table ServerParameters : cette table contient la liste de tous les paramètres propres au serveur de sécurité :
 - Le nom de login au service d'administration utilisé pour authentifier l'administrateur (*administratorLogin*);
 - Le mot de passe d'accès au service d'administration utilisé pour authentifier l'administrateur (*administratorPassword*);
 - L'adresse de connexion au serveur de fichiers (*fileServerLink*);
 - Le nom de login pour la connexion au serveur de fichiers (*fileServerLogin*);
 - Le mot de passe pour la connexion au serveur de fichiers (*fileServerPassword*);
 - Le répertoire dans lequel les fichiers sont stockés au niveau du serveur de fichiers (*fileServerApplicationFolder*);
 - Le délai de vérification des sessions invalides (*invalidSessionsVerificationDelay*). Cette vérification passe en revue toutes les sessions ouvertes par les utilisateurs et supprime celles dont la durée de validité est échuée.
 - Délai de validité des sessions (*sessionValidityDelay*). Ce délai spécifie le délai durant lequel une session d'utilisateur est valide.

15.2 Modèle relationnel

Le modèle relationnel de la base de données correspondant au modèle conceptuel ci-dessus est le suivant :

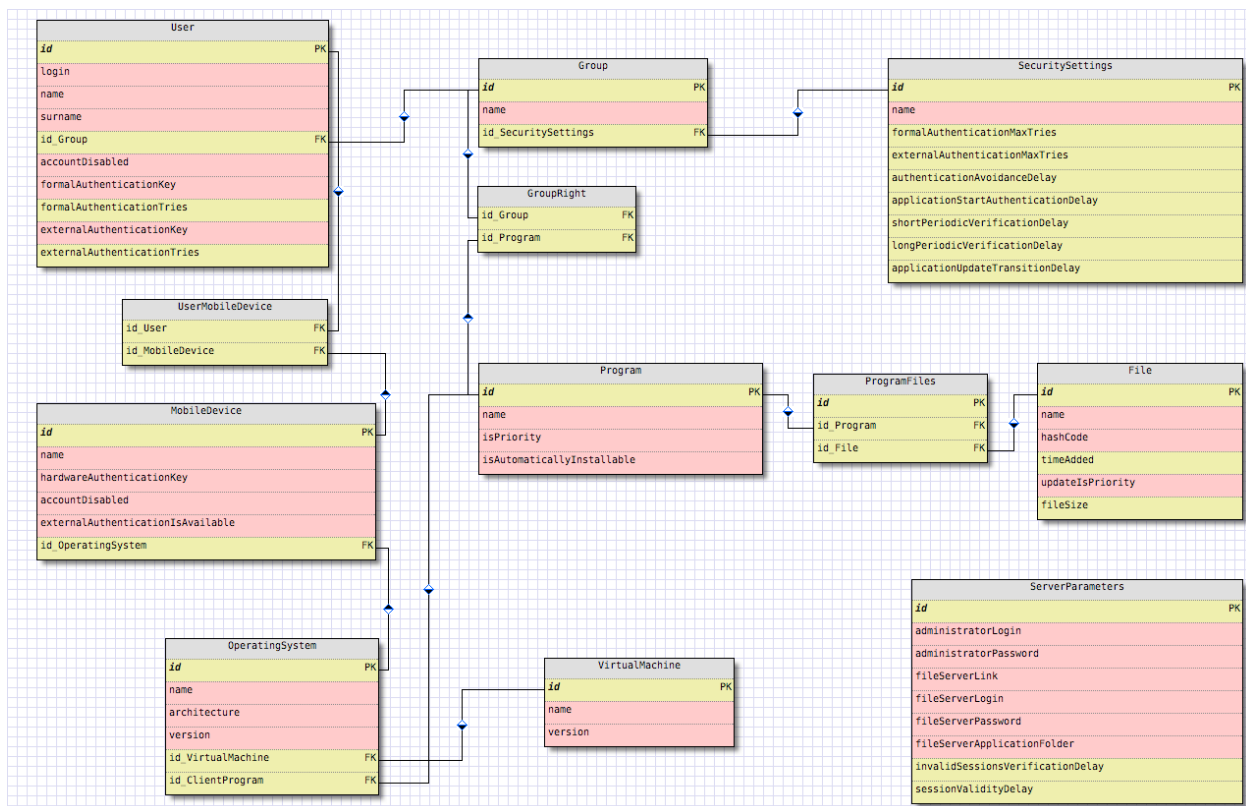


Figure 25: Schéma conceptuel de la base de données

Le schéma conceptuel est composé de douze tables. Les liens entre les tables dans le schéma conceptuel sont à présent représentés par de nouvelles tables dans le schéma relationnel. Nous trouvons ainsi les nouvelles tables :

- ◇ *UserMobileDevice* qui correspond au lien *uses*;
- ◇ *GroupRight* qui correspond au lien *is allowed to use*;
- ◇ *ProgramFiles* qui correspond au lien *contains*;

Les liens qui ne sont pas cités ci-dessus possédaient tous au moins une cardinalité 1-1 permettant d'ajouter directement une clé étrangère dans les tables déjà existantes. Aucune nouvelle table n'a donc été créée pour ces liens.

Au niveau des couleurs utilisées dans le schéma, le jaune représente des entiers et le rouge clair des chaînes de texte ou des booléens.

16. Diagrammes UML des classes

16.1 Classes du client lourd

Le diagramme des classes de la figure 26 représente les différents services du client lourd de sécurité. Dans un but de faciliter la lisibilité de ce schéma, seules les principales classes du client lourd sont représentées dans le diagramme, le but de ce dernier étant de montrer le regroupement des classes.

Au niveau du client, nous retrouvons cinq catégories principales :

- ◇ Le groupe noir contient les classes utilisées pour gérer les communications avec le serveur :
 - La classe *ClientRequests* contient un ensemble de méthodes Java utilisées par les différentes classes du client lourd pour communiquer avec le serveur. Ces méthodes génèrent les requêtes au format Java, les transmettent à la classe *ClientCommunications* afin qu'elles soient envoyées au serveur, puis extraient les informations des réponses Java qu'elles reçoivent afin de retourner le résultat demandé.
 - La classe *ClientCommunications* s'occupe de gestion des communications sécurisées directes avec le serveur. Elle effectue la conversion de la requête Java en un document XML à l'aide de la classe *XMLMessageParser*. Une fois le document XML créé, ce dernier est transmis au serveur, puis la réponse XML du serveur est traduite en réponse Java qui est retournée à la classe *ClientRequests*. Cette classe s'occupe également de demander et de recevoir les fichiers envoyés par le serveur.
 - La classe *XMLMessageParser* effectue la conversion entre les requêtes au format Java et les documents XML. Le parseur XML utilisé est le parseur kXML2.
- ◇ Le groupe bleu contient les classes implémentant le service de gestion des mises à jour :
 - La classe *ClientUpdateManager* gère le contrôle de la version et de l'intégrité des fichiers d'applications et les met à jour si nécessaire. Cette classe, qui est la première classe démarrée sur le client lourd, exécute le service d'authentification une fois les mises à jour terminées, ainsi que le chargeur d'applications (développé dans le cadre d'un autre travail de diplôme).
 - La classe *ClientUpdateGUI* est l'interface graphique principale du service de mise à jour permettant d'afficher le statut et l'avancement de ce service.
 - La classe *ClientUpdateApplicationGUI* offre une interface graphique permettant à l'utilisateur de sélectionner les applications à mettre à jour ou à installer.
- ◇ Le groupe rouge contient les classes gérant les authentifications et la sécurité en général :
 - La classe *ClientAuthenticationManager* est la classe centrale du client lourd. Elle est démarrée une fois que le processus de mise à jour est terminé et contient tous les paramètres globaux au client de sécurité, tels que les identifiants de sécurité (du client et des applications) ou le numéro IMEI de l'appareil mobile. Elle fait le lien entre toutes les classes, gère les interactions entre celles-ci et gère leur terminaison en cas de fermeture du client lourd. Elle gère en outre l'exécution des authentifications. Elle est enfin instanciée de manière unique, car elle centralise tous les paramètres de sécurité du client mobile.
 - La classe *ClientApplicationManager* permet de contrôler les applications désirant s'exécuter sur le client lourd et gère les demandes d'authentification en fonction des paramètres de sécurité imposés par le serveur et en fonction de l'état des applications. Lorsqu'une

authentification est nécessaire, cette classe appelle la classe *ClientAuthenticationManager* pour exécuter l'authentification requise.

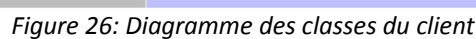
- La classe *ClientApplicationTimer* définit un *timer* (temporisateur) permettant de savoir quand les authentifications ont été faites et quand une nouvelle authentification est nécessaire. Pour ce faire, chaque instance de cette classe contient une date de création. Une méthode permet de savoir si ce temporisateur est échu en fonction d'un délai donné en paramètre qui est comparé à la date de création de l'instance. Cette classe est utilisée par la classe *ClientApplicationManager* pour stocker les temporisateurs des authentifications et pour ainsi effectuer une nouvelle authentification si l'un des délais est échu.
- ◇ Le groupe vert contient les services d'authentifications appelés par la classe *ClientAuthenticationManager* lorsqu'une authentification est nécessaire :
 - La classe *ClientAuthenticationServiceHardware* contient les méthodes nécessaires pour effectuer une authentification matérielle de l'appareil mobile en lui extrayant son numéro IMEI. Cette classe utilise la classe *ClientAuthenticationServiceHardwareGUI* comme interface graphique.
 - La classe *ClientAuthenticationServiceExternal* contient les méthodes nécessaires pour effectuer une authentification externe par Bluetooth. Cette classe utilise la classe *ClientAuthenticationServiceExternalGUI* comme interface graphique.
 - La classe *ClientAuthenticationServiceFormal* contient les méthodes nécessaires pour effectuer une authentification formelle de l'utilisateur en lui demandant son nom et son mot de passe. Elle utilise la classe *ClientAuthenticationServiceFormalGUI* comme interface graphique.
- ◇ Enfin, le groupe en jaune représente le module de sécurité faisant office de lien entre les applications et la classe *ClientAuthenticationManager*.

Les groupes représentent ainsi chacun une fonctionnalité bien distincte et offrent ainsi une relative indépendance entre ceux-ci.

Il est à noter que l'interface *ClientConfiguration*, non représentée dans ce diagramme, centralise toutes les constantes du client lourd. Cette interface est utilisée par la majorité des classes du client dans le but de faciliter la modification des constantes.

Remarque :

L'adresse du serveur de sécurité est stockée de manière fixe dans l'interface *ClientConfiguration*, car la modification de l'adresse par l'utilisateur n'est pas souhaitable et peut présenter un risque de sécurité, tel que la connexion à un serveur non autorisé.



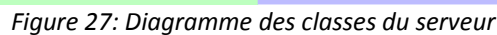
16.2 Classes du serveur

La figure 27 représente les différentes classes du serveur de sécurité. Tout comme le client, seules les classes principales sont représentées dans ce diagramme. Le serveur est décomposé en quatre groupes distincts :

- ◇ Le groupe en rouge représente le service de mise à jour. Nous y trouvons :
 - La servlet *ServletUpdate* permettant aux clients lourds de s'y connecter dans le but d'effectuer des demandes d'envoi de fichiers d'applications. Cette servlet utilise un parseur XML JAXB afin de traduire la requête XML en requête Java, puis transmet cette dernière au répartiteur de requêtes de mises à jour (*ServerUpdateRequestDispatcher*) et reçoit en retour le flux vers le fichier si la requête a été validée et acceptée. La servlet transmet ensuite le fichier au client.
 - La classe *ServerUpdateRequestDispatcher* permet de répartir les requêtes qui concernent le service de mise à jour au sein de celui-ci. De manière plus précise, cette classe contrôle le nom de la méthode demandée dans la requête, extrait les paramètres de la méthode puis redirige cette requête vers la méthode concernée dans le gestionnaire des mises à jour (*ServerUpdateManager*).
 - La classe *ServerUpdateManager* vérifie la légitimité de la requête (en vérifiant quel utilisateur l'a émise et si l'utilisateur est autorisé à recevoir le fichier qu'il demande). Dans le cas où la requête est légitime, cette classe fait appel à la classe *ServerFileAccess* afin de récupérer le flux vers le fichier depuis le serveur de fichiers, puis retourne ce flux aux classes des couches « inférieures » afin que la servlet puisse envoyer le fichier. Cette classe s'occupe aussi de l'ajout de nouveaux fichiers sur le serveur, ainsi que de la suppression des fichiers, et ce, de manière à ce que la base de données soit toujours à jour par rapport aux fichiers stockés sur le serveur.
- ◇ Le groupe en bleu représente le service de sécurité du serveur. La structure est très similaire au service de mise à jour. Nous y trouvons :
 - La servlet *ServletAuthentication* permet aux clients lourds d'effectuer des requêtes de sécurité destinées au serveur, telles que des demandes d'authentifications. Ces requêtes sont tout d'abord traduites en objets Java à l'aide d'un parseur XML JAXB, puis elles sont transmises sans prétraitement au répartiteur de requêtes (*ServerSecurityRequestDispatcher*). Les résultats des requêtes venant en retour au format Java sont traduits en XML par JAXB avant d'être envoyés aux clients.
 - La servlet *ServletServerAccess* est destinée aux serveurs d'applications afin que ceux-ci puissent envoyer des demandes de contrôle d'identifiants de sécurité d'applications. La servlet reçoit un identifiant de sécurité de type *appld...*, le transmet au répartiteur de requête afin que l'identifiant soit vérifié puis renvoie l'un des messages suivants au serveur d'applications ayant effectué la demande en fonction de la validité de l'identifiant :
 - L'identifiant est valide : *valid : userName*
où *userName* est le nom de l'utilisateur utilisant l'application.
 - L'identifiant est invalide : *error : invalid id*
 - Une erreur est survenue sur le serveur : *error : errorDescription*
où *errorDescription* est la description de l'erreur survenue sur le serveur.

- La classe *ServerSecurityRequestDispatcher* extrait le nom et les paramètres de la méthode appelée dans la requête, puis redirige la requête vers la méthode concernée dans le gestionnaire de sécurité *ServerSecurityManager*. Cette classe met alors en forme les réponses retournées par le gestionnaire de sécurité avant de les faire suivre à la servlet.
- La classe *ServerSecurityManager* est la classe centrale du serveur. Elle contient la liste de toutes les sessions ouvertes par les utilisateurs sur le serveur ainsi que toutes les méthodes liées à la sécurité, telles que les contrôles des authentifications des utilisateurs et des appareils mobiles, les contrôles des droits d'utilisation des applications, la demande des paramètres de sécurité, etc. Cette classe est instanciée de manière unique, car il ne doit exister qu'une seule liste des sessions ouvertes.
- La classe *SessionSecurityInformation* est utilisée par le gestionnaire de sécurité pour stocker les informations relatives aux sessions ouvertes par les utilisateurs. Chaque session contient le nom de login de l'utilisateur, le numéro IMEI de l'appareil mobile qu'il utilise et l'heure et la date de création de cette session. Ces sessions sont ensuite référées par l'identifiant de sécurité attribué au client lourd lors de la création de la session.
- La classe *ServerMaintenance* est une classe instanciée une seule fois lors du démarrage du serveur et qui périodiquement contrôle l'état de chaque session dans le but de supprimer les sessions devenues invalides.
- ◇ Le groupe en vert représente le service d'administration. Il est composé des classes suivantes :
 - La classe *ServerAdministration* comporte toutes les méthodes nécessaires au traitement des demandes d'administration provenant des pages JSP, telles que l'ajout, la modification ou la suppression d'utilisateurs, la gestion des paramètres de sécurité des clients lourds, la gestion des paramètres du serveur, etc. Cette classe représente le contrôleur du service d'administration, les vues étant les pages JSP non représentées dans ce schéma UML.
 - La servlet *ServletUpload* est utilisée pour recevoir de nouveaux fichiers à ajouter sur le serveur de fichiers. Une fois un nouveau fichier reçu, la servlet extrait les informations de ce fichier, telles que son nom, sa taille et son code haché, puis transmet ces informations à la classe de gestion des mises à jour (*ServerUpdateManager*) afin d'ajouter ou de modifier le fichier sur le serveur de fichiers.
- ◇ Le groupe en jaune représente le modèle du service, c'est-à-dire les classes d'accès à la base de données et au serveur de fichiers :
 - La classe *ServerFileAccess* permet au serveur de sécurité d'accéder au serveur de fichiers WebDAV afin de lire, ajouter ou supprimer des fichiers.
 - La classe *ServerDatabaseAccess* contient les méthodes d'accès à la base de données qui sont communes au service de mise à jour et au service de sécurité.
 - La classe *ServerDatabaseAccessSecurity* hérite de la classe *ServerFileAccess* et contient les méthodes d'accès à la base de données qui ne sont utilisées que par le service de sécurité.
 - La classe *ServerDatabaseAccessAdministration* hérite de la classe *ServerFileAccess* et contient les méthodes d'accès à la base de données qui ne sont utilisées que par le service d'administration.

Le serveur de sécurité est ainsi aussi clairement séparé en services distincts afin de les rendre aussi indépendants que possible. Il est à noter que, tout comme le schéma du client lourd, la classe *ServerConfiguration* (non représentée dans ce diagramme) centralise les constantes du serveur de sécurité.



17. Rétrospective

Le pré-projet de diplôme a permis d'établir un cahier des charges et une planification précise des tâches à effectuer dans le cadre de ce projet. En pratique, la réalisation d'un tel projet peut engendrer des problèmes inattendus requérant du temps supplémentaire, alors que d'autres tâches peuvent être raccourcies, voire supprimées au profit d'une autre solution. Il est ainsi intéressant, à la fin du projet, de faire un retour sur la planification et sur le cahier des charges pour voir quelles tâches se sont bien déroulées et quelles autres ont posé certains problèmes.

17.1 Retour sur la planification

17.1.1 Macro-calendrier

Le macro-calendrier est une manière très simple de représenter le déroulement d'un projet dans son ensemble. La figure 29 est une représentation théorique du déroulement du projet. Il est à noter que j'ai commencé la réalisation du projet deux semaines avant le début officiel, afin d'avoir une marge de sécurité dans le cas où un problème important venait à apparaître, ce qui n'a fort heureusement pas été le cas. J'ai ainsi profité de ce temps supplémentaire et de certains week-ends afin d'ajouter des fonctionnalités telles que l'authentification externe et afin d'effectuer un plus grand nombre de tests de l'application. La figure 28 représente le déroulement réel du projet.

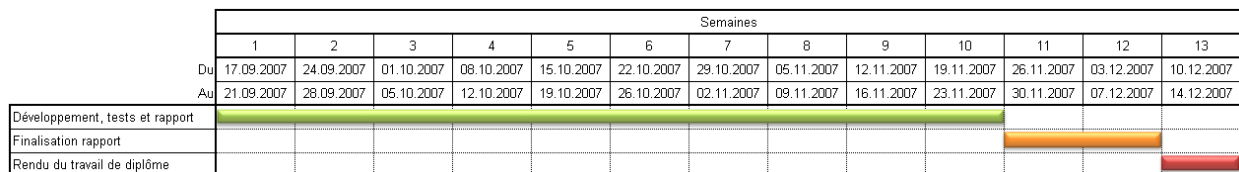


Figure 29: Macro-calendrier initial

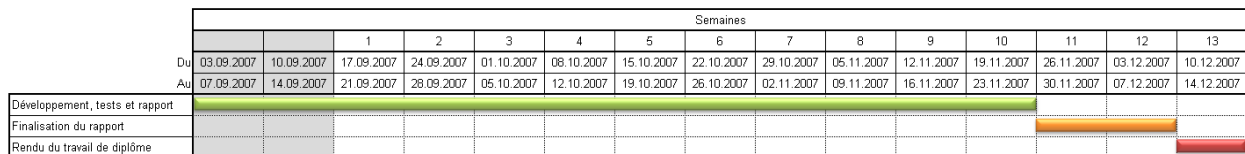


Figure 28: Macro-calendrier réel

17.1.2 Micro-calendrier

La planification des différentes tâches décrites dans le cahier des charges est représentée dans le micro-calendrier «Micro-calendrier – détails» à la page 73, qui représente une vue générale du déroulement du projet. Le calendrier 32 «Micro-calendrier – général» à la même page représente une vue détaillée (jour pour jour) du déroulement de ce projet. Ce calendrier m'a permis de surveiller mon avancement dans le but de savoir si j'avais du retard ou de l'avance tout au long du projet.

L'avancement réel, représenté dans la figure 30, diffère quelque peu de la planification en raison de certains problèmes rencontrés, en particulier au niveau de l'implémentation de la connexion sécurisée SSL et de l'exception *invalid padding* qu'il a fallu résoudre. La gestion des mises à jour a aussi requis un peu plus de temps en raison du contrôle nécessaire des dépendances entre les fichiers afin d'éviter des problèmes de compatibilité de versions lors des mises à jour. Le temps supplémentaire à disposition a en outre permis de réaliser l'authentification externe, d'effectuer un plus grand nombre de tests de l'application et d'approfondir les explications dans le rapport, ce qui explique le dépassement des durées planifiées. Quant aux absences, celles-ci sont dues au Jeûne fédéral, au forum HES-SO'07, à un entretien d'information dans une entreprise d'informatique et au recrutement militaire.

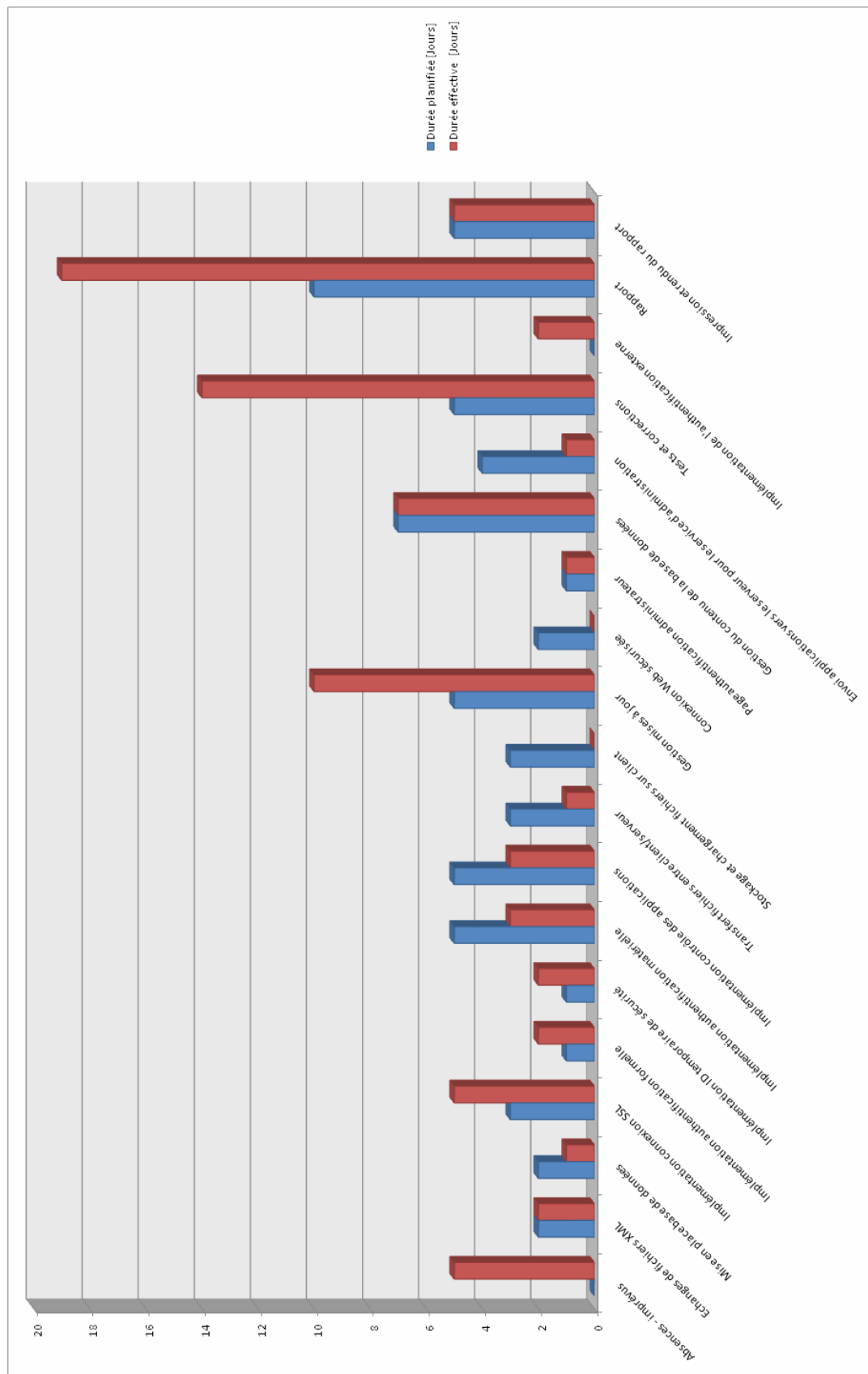


Figure 30: Graphique des durées de chaque tâche

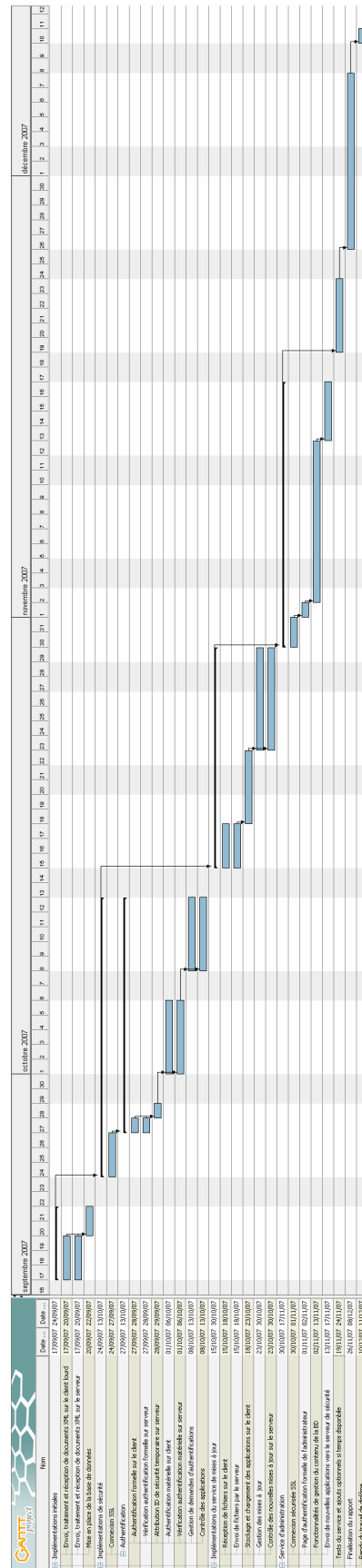


Figure 31: Micro-calendrier – détails

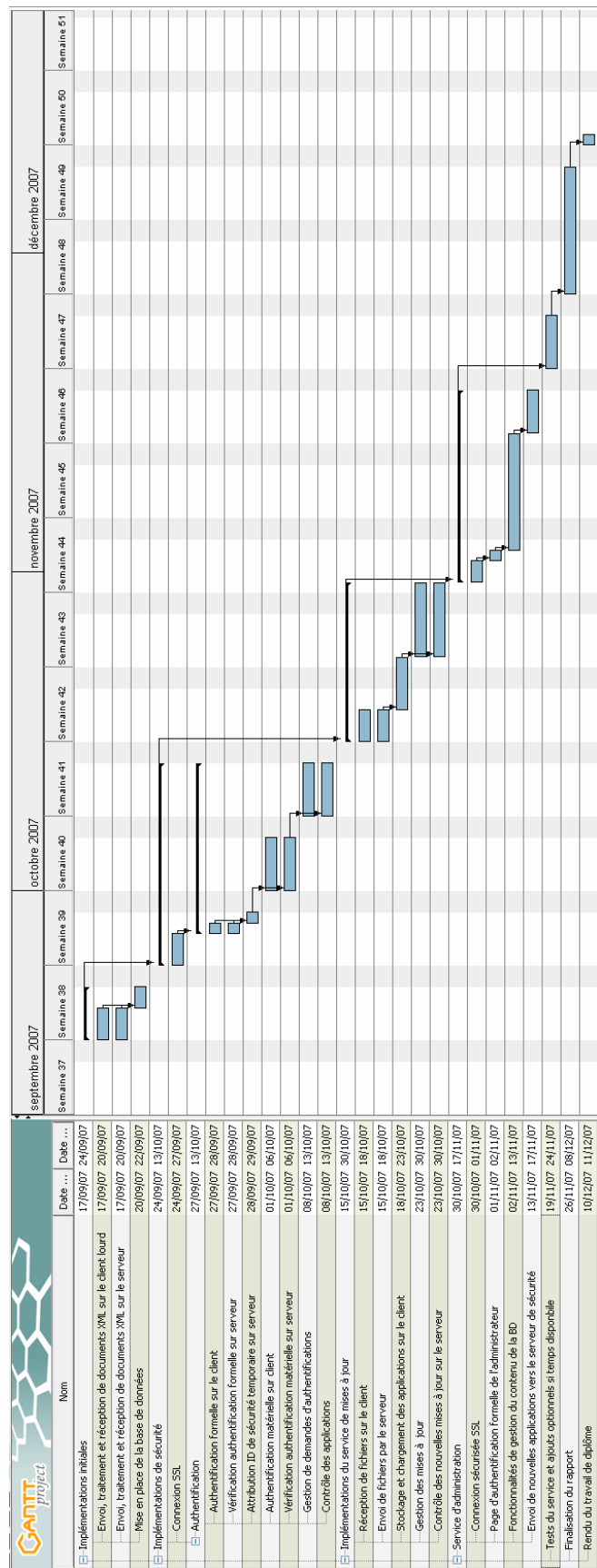


Figure 32: Micro-calendrier – général

17.2 Retour sur le cahier des charges

Les tâches prioritaires initialement prévues dans le cahier des charges ont toutes été réalisées, et ce, sans nécessiter de changement au niveau des solutions utilisées.

Au niveau des tâches optionnelles, la réalisation de l'interruption d'un téléchargement a également pu être réalisée sans problème particulier. En revanche, la possibilité de mettre à jour des applications en cours d'utilisation du client lourd est une fonctionnalité dont l'implémentation aurait posé un certain nombre d'obstacles qu'il n'aurait pas été possible de résoudre dans le cadre de ce travail de diplôme. En effet, contrôler une application en calculant son code haché est une opération qui nécessite une grande puissance de calcul, et un tel contrôle handicaperait d'autres applications en cours d'exécution. De plus, la mise à jour d'une application lorsqu'elle est démarrée n'est possible que par un redémarrage complet de l'application, voire du client lourd lui-même. Fort de ce constat, et au vu du temps relativement limité qu'il restait à disposition en fin de projet, cette tâche n'a pas pu être réalisée.

Enfin, la tâche d'implémentation d'un contrôle d'authentification biométrique a été abandonnée en cours de projet au profit d'une solution d'authentification externe par Bluetooth. L'exemple d'une authentification par empreinte digitale n'aurait en effet pas été utilisable en pratique en raison du manque de lecteurs biométriques compatibles. Après discussion avec M. Jaton, la solution d'authentification externe à l'aide d'un appareil Bluetooth semblait être une solution efficace qui pourrait facilement être adaptée aux appareils mobiles utilisés. C'est la raison pour laquelle l'authentification externe a été retenue et réalisée.

17.3 Améliorations possibles

Le service de sécurité et le service de mise à jour du client lourd sont actuellement totalement fonctionnels. Au niveau des fonctionnalités qu'il serait possible d'ajouter au client lourd, celles-ci dépendent de l'utilisation que l'on souhaite en faire. Dans le domaine médical, pour lequel le client a été étudié, il n'est pas nécessaire de lui ajouter des fonctionnalités pour pouvoir l'utiliser.

En revanche, des secteurs d'activités différents tels que le domaine bancaire ou financier peuvent requérir un niveau de sécurité plus important, notamment au niveau du stockage des applications et des données. En effet, les données enregistrées par le client lourd sur l'appareil mobile ne sont pas chiffrées ou protégées contre un accès indésirable. Bien que les fichiers d'applications soient protégés contre une quelconque atteinte à leur intégrité, ces fichiers pourraient être copiés, décompilés et analysés de manière à mieux comprendre leur fonctionnement dans le but de trouver des failles exploitables. Plus problématique encore, les informations personnelles stockées par les applications du client lourd, telles que les contacts ou les rendez-vous, ne sont pas chiffrées et peuvent être facilement copiées. Ce problème peut se présenter lorsque le PDA est réquisitionné dans les contrôles de douane par exemple, et peut nuire à la confidentialité des données. Une sécurité accrue est ainsi nécessaire.

Les solutions suivantes pourraient être envisagées pour protéger l'accès aux données confidentielles :

- ◇ La première solution, probablement la plus sûre, serait de garder toutes les informations sur le serveur et de ne stocker aucune donnée sur l'appareil mobile. Cette solution requiert toutefois une connexion constante avec le serveur et engendre des transferts importants de données, ce qui peut devenir problématique dans des environnements avec des réseaux à débit très faible.
- ◇ La seconde solution est basée sur le chiffrement de toutes les données à l'aide de clés asymétriques, sur une carte mémoire par exemple. Un tel chiffrement utiliserait une clé publique pour chiffrer les données et qui serait lisible et accessible à tous, et une clé privée stockée sur le serveur permettant de déchiffrer les données. Ainsi, toutes les données stockées sur le client seraient chiffrées à l'aide de la clé publique. En cas de réquisition du téléphone ou

du PDA, les données ne seraient pas lisibles et le seul moyen de les déchiffrer serait de s'authentifier sur le serveur afin de récupérer la clé privée. Malheureusement, l'algorithme de chiffrement asymétrique requiert une grande puissance de calcul, ce qui est peu adapté à des appareils mobiles.

- ◇ La solution suivante est une combinaison des deux premières solutions proposées. Cette solution serait basée sur un chiffrement des données, mais de manière symétrique, ce qui signifie que l'on utilise une même clé pour chiffrer et pour déchiffrer. Il faut alors s'assurer que cette clé reste confidentielle, en la transférant du serveur au client à chaque connexion de manière sécurisée et en ne la gardant qu'en mémoire vive par exemple. Le client lourd devrait alors rester ouvert afin de pouvoir déchiffrer les données, la fermeture de celui-ci signifiant la perte de la clé et donc l'impossibilité de lire les données. Dans un tel cas, une nouvelle connexion au serveur serait nécessaire pour obtenir à nouveau la clé de déchiffrement. La problématique principale serait alors le renouvellement périodique des clés afin de garder un bon niveau de sécurité.
- ◇ La dernière solution est basée sur une suppression de toutes les données confidentielles en cas de réquisition du PDA. Cette suppression serait activée manuellement par l'utilisateur ou lors d'une authentification erronée, si le PDA est éloigné du badge Bluetooth par exemple. Une fois l'utilisateur en « sécurité », celui-ci pourrait restaurer toutes ses données depuis le serveur. Cette solution pourrait aussi être combinée à la seconde ou la troisième solution qui consisterait à chiffrer les données en cas de danger, puis à transférer toutes les données sur une carte mémoire au lieu de les supprimer. Cela permettrait de restaurer ces données ultérieurement sans engendrer de gros transferts de données avec le serveur. Le chiffrement des données ne serait ainsi effectué qu'en cas de besoin, économisant la puissance de calcul le reste du temps. Au niveau pratique, la solution de suppression et de restauration des données est probablement la solution nécessitant le moins de modifications au niveau du client lourd. Dans le cadre du service de sécurité, il suffirait en effet d'effectuer une demande de suppression des informations confidentielles lorsqu'une authentification est erronée. La suppression à proprement dite devrait être effectuée par une application externe (jar) qui pourrait soit être exécutée par le client lourd, soit directement par l'utilisateur dans le cas où le client lourd ne serait pas en exécution. Enfin, le fait de ne pas chiffrer les données lorsque cela n'est pas nécessaire permet l'utilisation des applications du système d'exploitation pour accéder à ces données, comme le carnet d'adresses ou le gestionnaire des rendez-vous de Windows Mobile. L'utilisateur n'est donc pas contraint d'accéder à ces données qu'au travers du client lourd, ce qui est le cas de toutes les autres solutions proposées.

D'autres améliorations pourraient également être apportées au niveau du client lourd, telles l'accélération de l'authentification externe en recherchant l'adresse MAC du badge Bluetooth au lieu de rechercher toutes les adresses MAC disponibles. Le contrôle périodique de l'intégrité des applications en cours d'exécution pourrait également faire partie des améliorations possibles, ce contrôle ne s'effectuant actuellement qu'au démarrage du client lourd.

Enfin, au niveau du serveur de sécurité, une fonctionnalité intéressante serait d'ajouter un journal d'événements afin d'y répertorier toutes les erreurs qui sont survenues dans le but de permettre à l'administrateur d'en prendre connaissance et d'essayer de remédier à ces problèmes si nécessaire.

17.4 Problèmes connus

Certains problèmes décelés lors des tests n'ont pas pu être corrigés, car ceux-ci ne sont pas engendrés par des erreurs au niveau de l'application, mais par des erreurs au niveau des logiciels ou systèmes tiers utilisés. Ces problèmes sont rappelés ci-dessous :

- ◇ La mise en veille ou la désactivation du réseau sans fil du PDA lorsque celui-ci communique avec le serveur peut engendrer un blocage de la communication et du client lui-même. Le client lourd réagit toutefois encore aux actions de l'utilisateur, ce qui permet de le fermer si nécessaire¹⁵. Ce problème est dû à une non-détection de l'erreur par Java. En outre, en raison de l'impossibilité de mettre le PDA en veille, l'utilisation de programmes tiers pour le verrouillage du PDA est nécessaire afin d'éviter des actions imprévues si celui-ci se trouve dans un étui souple et qu'un objet entre en contact avec l'écran par exemple. Il n'a malheureusement pas encore été possible de trouver un programme satisfaisant permettant de résoudre ce problème.
- ◇ L'envoi sur le serveur de fichiers plus petits que 50 octets peut poser des problèmes avec les navigateurs Firefox et Safari lorsque la connexion utilisée est sécurisée. Les fichiers reçus par le serveur sont en effet vides. Ce problème n'apparaît toutefois pas avec Internet Explorer.¹⁶ Cette erreur est probablement due à un problème au niveau des tampons de réception des fichiers qui ne sont pas vidés correctement sur le serveur.
- ◇ Le client lourd ne fonctionne actuellement que sur Windows Mobile 5.0 en raison de l'utilisation de JNI et de bibliothèques DLL qui ne sont compatibles qu'avec le système de Microsoft. La portabilité de l'application n'est donc garantie que sur des systèmes compatibles avec ce type de bibliothèques. Ce problème est dû à l'impossibilité d'accéder aux ressources du système d'exploitation depuis Java directement, nécessitant ainsi l'utilisation de JNI pour les authentifications externes par Bluetooth et matérielles par la lecture du code IMEI.

L'utilisation de PDA ou de téléphones mobiles engendre par ailleurs des problèmes inhérents aux appareils mobiles. Parmi les différents problèmes qu'il est possible de rencontrer, le principal d'entre eux est la perte de la connexion au serveur lorsque l'utilisateur se déplace et qu'il entre dans un environnement sans réseau (tunnels, sous-sols, etc.). Le client lourd étant développé pour fonctionner avec un serveur, l'utilisateur ne pourrait pas continuer à utiliser le client lourd dans un tel environnement. Il se peut aussi que l'adresse IP de l'appareil mobile change en cours d'exécution si l'utilisateur se déplace et que le PDA se connecte à une autre borne Wi-Fi par exemple. Ce problème n'a en revanche pas de conséquences importantes sur le client lourd, car c'est ce dernier qui initie les connexions vers le serveur et non l'inverse.

Au niveau de la sécurité, un appareil mobile peut très facilement être volé, ce qui peut présenter un risque au niveau de la confidentialité des données. Les authentifications et autres mesures de sécurité implémentées dans le cadre de ce projet permettent de réduire les risques encourus dans un tel cas.

Enfin, le dernier problème principal concerne l'autonomie des appareils mobiles. En effet, l'utilisation de plusieurs technologies de communications sans fil simultanément, telles que la téléphonie, le Bluetooth et le Wi-Fi, diminue considérablement l'autonomie du PDA. Lors d'une telle utilisation, il faut s'attendre à une autonomie de quatre à huit heures au maximum avec les appareils mobiles actuels. Il serait donc nécessaire pour l'utilisateur du client lourd d'effectuer une recharge complète de la batterie régulièrement ou de posséder plusieurs batteries qu'il serait possible de changer en cours d'utilisation, ce qui peut s'avérer être très contraignant.

15 Pour plus d'informations, voir le chapitre 19.1.6, à la page 91.

16 Pour plus d'informations, voir le chapitre 19.1.5.8 à la page 88.

18. Conclusion

Le développement de ce projet a permis de soulever plusieurs points importants.

D'une part, la réalisation d'un projet de cette ampleur induit inévitablement divers obstacles, bien qu'aucun problème trop important ne soit survenu dans le cadre de ce projet. La principale difficulté a été rencontrée au moment de la création du système de mise à jour pour lequel la réalisation a été plus complexe que prévu. Les fonctionnalités supplémentaires et imprévues ne se sont finalement pas avérées trop problématiques à implémenter, car celles-ci ont pu être réalisées sans engendrer de décalage dans la planification. Cette difficulté a toutefois permis de rappeler qu'il est nécessaire de garder des marges de sécurité lors de l'établissement des délais dans la planification, en prévision aux difficultés pouvant apparaître de manière inattendue.

D'autre part, le travail en équipe a été très important pour réaliser ce projet. Il a été indispensable d'établir des concepts et des schémas théoriques solides en discutant en groupe avant de commencer la réalisation pratique de l'application, afin que l'intégration des diverses parties de celle-ci en fin de projet puisse être effectuée facilement. Cette étude théorique a été réalisée au moment du pré-projet de diplôme et s'est avérée très utile. De plus, la collaboration au sein de l'équipe avec MM. Wulliamoz et Probst s'est extrêmement bien passée et a été très instructive, tant du point de vue professionnelle que personnelle. En fin de projet, les essais d'assemblage des divers services ont pu se faire rapidement et très facilement, ce qui indique que le travail en équipe a été un succès.

Enfin, la notion de sécurité des applications, qui de nos jours semble être une caractéristique usuelle et normale aux yeux de la majorité des utilisateurs, n'en est pas moins une réelle difficulté à assurer pour les développeurs. En effet, lors de la réalisation d'une application, il est essentiel en tant que concepteur d'imaginer toutes les attaques et toutes les failles qu'il serait possible de rencontrer et d'exploiter dans les applications. Cette réflexion est loin d'être triviale, car il est nécessaire de se mettre à la place d'une personne malintentionnée et de chercher la manière avec laquelle on arriverait alors à passer outre les protections que l'on vient de mettre au point. Le risque majeur dans cette situation n'est pas dû au fait de trouver de nouvelles failles, car celles-ci pourraient alors être corrigées, mais le réel danger serait de ne pas les déceler, auquel cas celles-ci risqueraient d'être découvertes plus tard par une personne réellement malveillante. C'est la raison pour laquelle il n'est jamais possible de garantir l'immunité complète d'un système de sécurité, bien que le client lourd et le serveur développés durant ce projet offrent une bonne protection contre les différentes attaques possibles.

Je conclurais en disant que la réalisation de ce projet a été passionnante, le domaine de la sécurité sur des appareils mobiles étant encore peu exploré. Il a été possible de trouver diverses solutions afin d'obtenir une application fonctionnelle en fin de projet, même si le développement sur une telle plateforme n'a pas toujours été aisé en raison de fonctionnalités parfois manquantes sur celle-ci.

Il n'en reste pas moins que la sécurité reste un domaine extrêmement vaste et complexe, le développement des nouvelles technologies lui assurant avec certitude un très bel avenir.

Yverdon-Les-Bains, le 12.12.2007,

Stephen Badan.

19. Annexes

19.1 Tests

19.1.1 Remarques générales

Tous les tests qui suivent ont été réalisés avec les configurations suivantes :

- ◇ Client mobile :
 - HTC P3300 :
 - Windows Mobile 5.1 build 342
 - Processeur ARM cadencé à 200 MHz
 - Utilisation du Wi-Fi sur un réseau sans fil chiffré à l'aide du protocole WPA2 (*Wi-Fi Protected Access*) et du Bluetooth.
 - Machine virtuelle IBM J9 PPRO 11
 - Client lourd version 1.0
- ◇ Serveur :
 - Dell Optiplex GX620 :
 - Processeur Pentium D cadencé à 3 GHz
 - 1 Gb de RAM
 - Réseau Ethernet 100 Mb/s
 - Windows XP Professionnel Service Pack 2 (version 5.1.2600)
 - Apache Tomcat version 5.0.28
 - Machine virtuelle Java Sun 1.6.0_03-b05

19.1.2 Test du service de gestion des demandes d'authentification

Les tests suivants ont été réalisés avec les délais donnés ci-dessous :

- ◇ Délai de démarrage d'applications : 40 secondes
- ◇ Délai d'évitement d'authentification : 30 secondes
- ◇ Long délai périodique d'authentification : 140 secondes
- ◇ Court délai périodique d'authentification : 60 secondes

No	Test effectué	Résultat attendu	Résultat observé	Remarque
1	Démarrer le client lourd sans démarrer d'application en répondant à toutes les authentifications de manière valide	Une fois le client lourd démarré, demande d'authentification externe après 60 secondes et après environ 130 secondes (en fonction du temps qu'a pris la première authentification externe), puis authentification matérielle et formelle après 140 secondes.	Idem au résultat attendu : ok	

		Le processus doit se répéter après l'authentification formelle et matérielle.		
2	Démarrer le client lourd sans application en choisissant d'éviter l'authentification formelle après 140 secondes	L'authentification formelle doit se fermer et être remplacée par une authentification externe. Une fois l'authentification externe effectuée, l'authentification formelle doit être effectuée à nouveau après 30 secondes. Cette nouvelle authentification ne doit pas pouvoir être contournée. Les contrôles périodiques vus au test 1 doivent ensuite s'effectuer normalement.	Idem au résultat attendu : ok	
3	Dans les options de sécurité de l'appareil mobile (dans la base de données), désactiver l'authentification externe sur l'appareil mobile et effectuer le test 1	Les authentifications externes doivent être remplacées par des authentifications formelles et il ne doit plus être possible d'effectuer un évitement d'authentification formelle.	Idem au résultat attendu : ok	
4	Rétablir les paramètres. Dans les options de sécurité du groupe de l'utilisateur, désactiver les évitements d'authentifications formelles	Les authentifications formelles ne doivent plus être évitables.	Idem au résultat attendu : ok	
5	En utilisant l'application de simulation d'applications, démarrer une application dès l'ouverture de l'interface graphique de l'application	Comme l'application est ouverte dans le délai de démarrage des applications, aucune authentification n'est requise.	Idem au résultat attendu : ok	
6	Idem au test précédent, mais ouvrir une application après le délai d'ouverture des applications, soit après 40 secondes.	Une authentification externe est effectuée avant que la nouvelle application ne puisse effectuer de requête	Idem au résultat attendu : ok	
7	A la suite du test 6, ouvrir une seconde application après le délai de démarrage d'applications	Aucune authentification n'est cette fois requise, car le délai de démarrage a été réinitialisé après l'authentification externe.	Idem au résultat attendu : ok	
8	Démarrer une application peu après une authentification externe de contrôle périodique à court terme	Comme l'authentification externe vient d'être exécutée, le délai de démarrage des applications est réinitialisé et aucune nouvelle authentification n'est requise à ce moment-là.	Idem au résultat attendu : ok	
9	Idem au test 9, mais démarrer une nouvelle application 40 secondes après une authentification externe provenant d'un contrôle périodique court	Une authentification externe est requise pour l'ouverture de la nouvelle application, car le délai de démarrage des applications est échoué	Idem au résultat attendu : ok	
10	Démarrer plusieurs applications simultanément et contrôler qu'elles soient mise en attente lors d'une demande d'authentification	Durant une authentification, toutes les demandes d'identifiants de sécurité sont mises en attente.	Idem au résultat attendu : ok	

19.1.3 Test des services d'authentification

Les tests suivants ont été réalisés avec les délais donnés ci-dessous :

- ◇ Nombre maximal d'authentifications formelles erronées : 3
- ◇ Nombre maximal d'authentifications externes erronées : 5

No	Test effectué	Résultat attendu	Résultat observé	Remarque
1	Lors d'une authentification formelle, saisir 3 mots de passe erronés	Une fenêtre d'erreur doit être affichée indiquant que le compte de l'utilisateur a été verrouillé	Idem au résultat attendu : ok	
2	Lors d'une authentification formelle, saisir 2 mots de passe erronés, puis le mot de passe correct. A l'authentification formelle suivante, saisir 2 mots de passe erronés puis le mot de passe correct.	Un mot de passe valide remet à zéro le compteur du nombre de mots de passe erronés saisi. Ainsi, aucune erreur ne doit apparaître lors du test.	Idem au résultat attendu : ok	
3	S'éloigner de l'appareil dont l'adresse MAC Bluetooth sert d'authentification externe et attendre qu'une authentification externe s'effectue	L'authentification externe doit échouer et le compte de l'utilisateur doit être bloqué.	Idem au résultat attendu : ok	
4	Démarrer le client lourd sans démarrer d'application en répondant à toutes les authentifications de démarrage de manière valide. Une fois le client démarré, redémarrer le serveur afin d'effacer tous les identifiants de sécurité sur le serveur.	Une fenêtre d'erreur doit être affichée sur le client lors de l'authentification ou du contrôle des applications, car l'identifiant de sécurité de l'utilisateur est n'existe pas sur le serveur, en raison de son redémarrage. Comme l'identifiant n'existe plus, l'accès au serveur est révoqué pour le client lourd et une exception est levée.	Idem au résultat attendu : ok	
5	Démarrer le client lourd en répondant à toutes les authentifications de démarrage de manière valide et en ne démarrant aucune application. Une fois le client démarré, bloquer le compte de l'utilisateur au niveau du serveur.	A la prochaine authentification, une erreur doit s'afficher sur le client lourd indiquant à l'utilisateur qu'il n'est plus autorisé à utiliser son compte. Le client lourd doit ensuite se fermer automatiquement.	Idem au résultat attendu : ok	L'état du compte de l'utilisateur n'est contrôlé que lors des authentifications.
6	Démarrer le client lourd en répondant à toutes les authentifications de démarrage de manière valide et en ne démarrant quelques applications. Une fois le client démarré, bloquer le compte de l'utilisateur au niveau du serveur.	Idem au test 5.	Idem au résultat attendu : ok	
7	Effectuer à nouveau le test 5, mais en bloquant l'appareil mobile.	A la prochaine authentification, une erreur doit s'afficher sur le client lourd indiquant que l'appareil mobile n'est plus autorisé à se connecter au service. Le client lourd doit ensuite se fermer	Idem au résultat attendu : ok	

		automatiquement,		
8	Effectuer à nouveau le test 6, mais en bloquant l'appareil mobile.	Idem au test 8.	Idem au résultat attendu : ok	

19.1.4 Tests du service de mise à jour

Les tests suivants ont été réalisés avec les programmes attribués au groupe de l'utilisateur de la manière suivante :

- ◇ Client, lié au système d'exploitation utilisé par l'utilisateur
 - Client.jar
 - mobileDeviceServices.dll
- ◇ Pack 1, prioritaire, installable automatiquement
 - Loader.jar
- ◇ Pack 2, non prioritaire, installable automatiquement
 - Application1.jar
 - commonDll.dll
- ◇ Pack 3, non prioritaire, installable automatiquement
 - Application2.jar
 - commonDll.dll

Toutes les applications données ci-dessus ne sont pas encore installées sur le client. Par ailleurs, la version du client et de la machine virtuelle est à jour au début des tests. Enfin, le délai de transition de mise à jour est suffisant élevé pour éviter toute mise à jour obligatoire.

No	Test effectué	Résultat attendu	Résultat observé	Remarque
1	Démarrer le client lourd	Le client vérifie la version de la machine virtuelle puis télécharge le fichier mobileDeviceServices.dll. Le client étant à jour, l'authentification matérielle, puis formelle doivent s'exécuter. Une fois l'utilisateur authentifié, l'application Loader.jar du Pack 1 doit être téléchargée automatiquement sans laisser le choix d'annuler le transfert. A la fin du téléchargement, une fenêtre de choix des mises à jour pour les applications Pack 2 et Pack 3 doit être ouverte.	Idem au résultat attendu.	
2	A la suite du test 1, sélectionner toutes les mises à jour, puis valider	Les applications Application1.jar, commonDll.dll et Application2.jar doivent être téléchargées. Une fois les applications téléchargées, la	Idem au résultat attendu.	Le client ne télécharge qu'une seule fois le fichier Application.jar, car il est partagé entre le Pack 2 et le Pack 3.

		fenêtre de simulation des programmes doit afficher les fichiers Application1.jar, Application2.jar, Loader.jar et commonDll.dll.		
3	Fermer l'application et la redémarrer.	Aucune mise à jour ne doit être téléchargée cette fois-ci, car tous les fichiers sont à jour.	Idem au résultat attendu.	
4	Fermer l'application, supprimer le fichier commonDll.dll, puis redémarrer le client.	Aucune mise à jour prioritaire ne doit être effectuée, puis la fenêtre de choix doit proposer l'installation des applications Pack 2 et Pack 3.	Idem au résultat attendu.	Comme le fichier commonDll.dll est partagé entre Pack 2 et Pack 3, les deux packs sont proposés pour l'installation, car ils sont tous deux incomplets.
5	A la suite du test 4, sélectionner les deux packs, puis valider.	Le fichier commonDll.dll doit être téléchargé une seule fois, puis la fenêtre de test des applications doit s'ouvrir avec les applications Loader.jar, Application1.jar, commonDll.dll et Application2.jar	Idem au résultat attendu.	Le fichier commonDll.dll étant commun aux deux applications, le client ne le télécharge qu'une seule fois.
6	Supprimer les fichiers commonDll.dll et Application2.jar, puis démarrer le client lourd. Sélectionner l'installation du Pack 3 uniquement, puis annuler le téléchargement du fichier commonDll.dll.	L'annulation du téléchargement du fichier commonDll.dll doit afficher un message d'erreur, puis engendrer la suppression du fichier Application2.jar si ce fichier a été téléchargé, car le Pack 3 est incomplet. Le fichier Application1.jar, appartenant au Pack 2, est supprimé aussi, car le fichier commonDll.dll est manquant, ce qui rend l'utilisation du fichier Application1.jar impossible. Au final, la fenêtre de simulation des applications ne doit afficher que le fichier Loader.jar.	Idem au résultat attendu.	
7	Relancer le client et remettre à jour tous les fichiers, puis fermer le client. Effectuer à nouveau le test 6, mais en sélectionnant cette fois-ci la mise à jour des deux packs. Annuler le téléchargement du fichier commonDll.dll (une fois seulement), puis laisser la mise à jour se terminer.	Un message d'avertissement doit s'afficher, puis le fichier commonDll.dll doit à nouveau commencer à être téléchargé, car ce second téléchargement fait partie de la mise à jour de l'autre pack sélectionné. Au final, dans la fenêtre de simulation, les fichiers ApplicationX.jar (où X est le numéro du Pack correctement téléchargé), commonDll.dll et Loader.jar doivent être présents dans la liste des applications.	Idem au résultat attendu.	Le fichier d'application du Pack dont le téléchargement a été annulé a été supprimé. L'annulation du téléchargement commonDll.dll n'a en revanche pas engendré la suppression du jar de l'autre Pack, car la DLL a à nouveau été téléchargée lors de la mise à jour de ce second pack.
8	Fermer le client, puis remplacer le fichier commonDll.dll par un fichier différent portant le même nom. Faire de même pour le fichier Application2.jar.	Après le téléchargement du fichier commonDll.dll, un message doit apparaître indiquant à l'utilisateur qu'un fichier partagé a été mis à jour	Idem au résultat attendu.	Afin d'éviter des problèmes de compatibilité, une application est mise à jour si l'un de ses

	Dans la fenêtre des demandes de mise à jour, sélectionner le Pack 2 uniquement.	par une autre application et que cela entraîne une mise à jour du Pack 3, car ce dernier utilise aussi ce fichier partagé. Au final, les fichiers commonDll.dll et Application2.jar sont téléchargés et tous les fichiers sont disponibles.		fichiers a été mis à jour lors de la mise à jour d'une autre application, et ce, même si l'utilisateur n'a pas sélectionné cette application. Un tel processus ne s'exécute que si l'application était déjà existante, mais ne s'applique pas aux applications non installées.
9	Réitérer le test 8, mais en modifiant au préalable le paramètre d'installation automatique à faux dans les paramètres de l'application pour le pack 3.	Etant donné que le pack 3 ne peut pas être mis à jour automatiquement, alors que le fichier commonDll.dll dont il dépend est modifié, les fichiers du pack 3 sont supprimés afin d'éviter tout problème. Au final, il reste les fichiers Application1.jar, commonDll.dll et Loader.jar dans la liste des fichiers disponibles.	Idem au résultat attendu.	
10	Restaurer la valeur d'installation automatique à vrai dans les paramètres du pack 3 et restaurer tous les fichiers. Ensuite, modifier les paramètres suivants pour les applications ci-dessous : Pack 1 : installation prioritaire = faux Pack 3 : installation prioritaire = vrai Remplacer le fichier commonDll.dll par un fichier différent de même nom et redémarrer le client. Faire de même pour le fichier Application1.jar. Dans la fenêtre des mises à jour facultatives, ne sélectionner aucune mise à jour et valider.	Lors des mises à jour des applications critiques, le fichier commonDll.dll est automatiquement mis à jour, sans laisser à l'utilisateur le choix d'annuler le téléchargement. Si l'utilisateur ne sélectionne pas le Pack 2, le client va tout de même contrôler les dépendances des fichiers. Comme l'un des fichiers du pack 2 a été modifié lors du téléchargement du Pack 3, un message d'erreur s'affiche et le Pack 2 est mis à jour.	Idem au résultat attendu : ok	Si l'utilisateur décidait d'annuler le téléchargement du Pack 2, alors le fichier Application1.jar serait supprimé.
11	Effectuer à nouveau le test 10, mais en indiquant cette fois-ci que le Pack 2 n'est pas installable automatiquement.	Le fichier commonDll.dll doit être téléchargé automatiquement. Comme la Pack 2 n'est ensuite pas sélectionné, un message d'avertissement s'affiche indiquant que le Pack 2 doit être mis à jour. Comme le Pack 2 n'est pas téléchargeable automatiquement, le Pack 2 est supprimé afin d'éviter toute erreur de compatibilité.	Idem au résultat attendu.	
12	Restaurer les paramètres de base. Copier et démarrer une version plus ancienne du client lourd.	Le client doit télécharger la dernière version du client automatiquement, sans laisser le choix d'annulation à	Idem au résultat attendu.	

		l'utilisateur. A la fin du téléchargement, un message d'avertissement de mise à jour doit être affiché, puis le fichier d'application du client doit être remplacé et l'application doit se terminer. Aucun fichier temporaire ne doit être présent à la fin de la mise à jour.		
13	Modifier la version de la machine virtuelle Java pour le système d'exploitation utilisé au niveau du serveur.	Un message d'erreur doit s'afficher sur le client indiquant à l'utilisateur que la machine virtuelle n'est pas à jour et qu'elle nécessite une réinstallation auprès de l'administrateur.	Idem au résultat attendu.	
14	Restaurer la valeur de version de la machine virtuelle, puis modifier la version du système d'exploitation utilisé au niveau de la base de données.	Un message d'erreur doit être affiché indiquant que le système d'exploitation n'a pas été trouvé pour le système d'exploitation spécifié.	Idem au résultat attendu.	
15	Restaurer la version de la machine virtuelle dans la base de données, puis remplacer le fichier mobileDeviceServices.dll dans l'appareil mobile par un fichier quelconque de même nom.	Le client mobile doit détecter que le code haché du faux fichier mobileDeviceServices.dll est incorrect et doit télécharger la version valide depuis le serveur automatiquement, sans laisser de choix à l'utilisateur. A la fin du téléchargement, l'authentification matérielle doit s'effectuer sans erreur.	Idem au résultat attendu.	
16	Dans le répertoire où est installé le programme, ajouter quelques fichiers divers, puis démarrer le client et le refermer de manière normale (sans créer d'exception).	Tous les fichiers ajoutés et sans lien avec l'application sont supprimés à la fermeture de l'application.	Idem au résultat attendu.	Les fichiers *.lnk ne sont pas supprimés, car ceux-ci peuvent être utilisés pour ouvrir l'application et leur suppression pourrait rendre impossible une nouvelle ouverture du client.
17	Créer un compte pour un utilisateur pour lequel aucune application n'est autorisée. Se connecter ensuite à ce compte, en s'étant préalablement connecté avec un compte possédant plusieurs applications.	Aucune application ne doit être mise à jour et la liste des applications disponibles dans la fenêtre des tests d'applications doit être vide. Après la fermeture normale du client, aucun fichier des anciennes applications ne doit être présent.	Idem au résultat attendu.	
18	Ajouter une application prioritaire non installable automatiquement pour le compte courant et démarrer le client avec ce compte.	Une erreur doit apparaître, informant l'utilisateur qu'une mise à jour critique doit être effectuée par l'administrateur et que l'appareil mobile n'est pas autorisé à se connecter jusqu'à ce que la mise à jour soit faite. Le programme doit ensuite se fermer.	Idem au résultat attendu.	
19	Restaurer les paramètres et réinstaller toutes les applications. Remplacer le	Dans la fenêtre des choix des mises à jour, l'application Pack 2 ne doit pas être décochable, car	Idem au résultat attendu.	Le téléchargement du fichier peut être supprimé comme

	fichier Application1.jar par un fichier différent de même nom et paramétrer la mise à jour de ce fichier comme étant prioritaire sur le serveur.	la mise à jour de cette application devient prioritaire en raison du fichier prioritaire Application1.jar qu'elle contient. La mise à jour de ce fichier doit donc se faire.		n'importe quelle autre application non prioritaire.
20	Restaurer les paramètres sur le serveur, puis modifier le délai de transition des mises à jour des fichiers pour le groupe de l'utilisateur courant afin que ce délai soit inférieur au temps écoulé depuis la dernière mise à jour du fichier sur le serveur.	Idem au résultat attendu du test 19.	Idem au résultat attendu.	
21	Restaurer les paramètres sur le serveur, puis interdire le délai de transition des mises à jour des fichiers pour le groupe de l'utilisateur courant sur le serveur.	Idem au résultat attendu du test 19.	Idem au résultat attendu.	
22	A la suite du test 21, supprimer le fichier Application1.jar et redémarrer le client lourd.	Aucune installation ou mise à jour ne doit être obligatoire cette fois-ci, car le fichier Application1.jar n'est pas installé sur le client lourd. L'utilisateur a donc le choix d'installer ou non l'application Pack 2.	Idem au résultat attendu.	

19.1.5 Tests du service d'administration

19.1.5.1 Tests généraux

No	Test effectué	Résultat attendu	Résultat observé	Remarque
1	S'authentifier sur le site avec un nom d'utilisateur ou un mot de passe erroné.	L'accès au service doit être refusé.	Idem au résultat attendu.	
2	S'authentifier sur le site avec un nom d'utilisateur et un mot de passe valide.	L'accès au service doit être accepté.	Idem au résultat attendu.	
3	Se déconnecter du site avec le lien « Log Out » présent dans le menu de droite.	La page courante doit être remplacée par un message d'erreur et le formulaire d'authentification doit à nouveau être affiché.	Idem au résultat attendu.	
4	En étant déconnecté, essayer d'ouvrir une page en la tapant directement dans la barre d'adresse du navigateur Web, par exemple la page « User.jsp ».	La page ne doit pas être affichée et doit être remplacée par un message d'erreur.	Idem au résultat attendu.	

19.1.5.2 Tests de la gestion des clients

Tous les tests suivants s'effectuent lorsque l'administrateur est authentifié.

No	Test effectué	Résultat attendu	Résultat observé	Remarque
----	---------------	------------------	------------------	----------

1	Créer un nouvel utilisateur possédant un nom de login d'un utilisateur déjà existant.	Un message d'erreur doit être affiché indiquant que le nom de login est déjà utilisé.	Idem au résultat attendu.	Lors de la création d'un nouvel utilisateur, le mot de passe choisi pour l'utilisateur est automatiquement haché. Si une modification est effectuée sur le mot de passe ultérieurement, le nouveau mot de passe est aussi haché.
2	Créer un nouvel utilisateur avec un nouveau nom de login.	L'utilisateur doit être créé.	Idem au résultat attendu.	
3	Choisir de modifier l'utilisateur puis modifier certains champs et cliquer sur « cancel modifications ».	Les valeurs des champs doivent être restaurées à leur valeur d'origine.	Idem au résultat attendu.	
4	Vérifier que le contrôle des champs s'effectue correctement pour chaque champ du formulaire.	Une erreur doit être affichée si un champ est vide ou invalide, indiquant quel est le champ concerné.	Idem au résultat attendu.	
5	Supprimer un utilisateur.	L'utilisateur doit être supprimé, ainsi que toutes les attributions d'appareils mobiles de cet utilisateur.	Idem au résultat attendu.	

19.1.5.3 Tests de la gestion des appareils mobiles

Tous les tests suivants s'effectuent lorsque l'administrateur est authentifié.

No	Test effectué	Résultat attendu	Résultat observé	Remarque
1	Créer un nouvel appareil mobile possédant un code IMEI d'un appareil mobile déjà existant.	Un message d'erreur doit être affiché indiquant que le code IMEI est déjà utilisé.	Idem au résultat attendu.	
2	Créer un nouvel appareil mobile possédant un nouveau code IMEI.	L'appareil mobile doit être créé.	Idem au résultat attendu.	
3	Choisir de modifier l'appareil puis modifier certains champs et cliquer sur « cancel modifications ».	Les valeurs des champs doivent être restaurées à leur valeur d'origine.	Idem au résultat attendu.	
4	Vérifier que le contrôle des champs s'effectue correctement pour chaque champ du formulaire.	Une erreur doit être affichée si un champ est vide ou invalide, indiquant quel est le champ concerné.	Idem au résultat attendu.	
5	Supprimer un appareil mobile.	L'appareil mobile doit être supprimé, ainsi que toutes les attributions de cet appareil mobile aux utilisateurs.	Idem au résultat attendu.	

19.1.5.4 Tests de la gestion des attributions des appareils mobiles

Tous les tests suivants s'effectuent lorsque l'administrateur est authentifié.

No	Test effectué	Résultat attendu	Résultat observé	Remarque
1	Sélectionner un utilisateur et essayer de lui ajouter et de lui	Les appareils mobiles doivent être ajoutés et supprimés de	Idem au résultat attendu.	Si aucun appareil mobile n'est

supprimer des appareils mobiles.	l'utilisateur en fonction du choix de l'utilisateur.	sélectionné et que l'utilisateur décide d'ajouter ou de supprimer un appareil, un message d'erreur est affiché.
----------------------------------	--	---

19.1.5.5 Tests de la gestion des machines virtuelles

Tous les tests suivants s'effectuent lorsque l'administrateur est authentifié.

No	Test effectué	Résultat attendu	Résultat observé	Remarque
1	Créer une nouvelle machine virtuelle possédant un nom et une version identiques à une machine virtuelle déjà existante	Un message d'erreur doit être affiché indiquant qu'une machine virtuelle avec ces paramètres existe déjà.	Idem au résultat attendu.	
2	Créer une nouvelle machine virtuelle avec un nouveau nom ou une nouvelle version	La machine virtuelle doit être créée.	Idem au résultat attendu.	
3	Choisir de modifier la machine virtuelle, puis modifier certains champs et cliquer sur « cancel modifications ».	Les valeurs des champs doivent être restaurées à leur valeur d'origine.	Idem au résultat attendu.	
4	Vérifier que le contrôle des champs s'effectue correctement pour chaque champ du formulaire.	Une erreur doit être affichée si un champ est vide ou invalide, indiquant quel est le champ concerné.	Idem au résultat attendu.	
5	Supprimer une machine virtuelle qu'un ou plusieurs systèmes d'exploitation utilisent.	Une erreur doit être affichée indiquant le nom des systèmes d'exploitation qui doivent être supprimés avant de pouvoir supprimer la machine virtuelle.	Idem au résultat attendu.	
6	Supprimer une machine virtuelle qui n'est utilisée par aucun système d'exploitation	La machine virtuelle doit être supprimée.	Idem au résultat attendu.	

19.1.5.6 Tests de la gestion des systèmes d'exploitation

Tous les tests suivants s'effectuent lorsque l'administrateur est authentifié.

No	Test effectué	Résultat attendu	Résultat observé	Remarque
1	Créer un nouveau système d'exploitation possédant un nom, une architecture et une version déjà existants.	Un message d'erreur doit être affiché indiquant qu'un système d'exploitation avec ces paramètres existe déjà.	Idem au résultat attendu.	
2	Créer un nouveau système d'exploitation avec un nouveau nom, une nouvelle version ou une nouvelle architecture.	Le système d'exploitation doit être créé.	Idem au résultat attendu.	
3	Choisir de modifier le système d'exploitation puis modifier certains champs et cliquer sur « cancel modifications ».	Les valeurs des champs doivent être restaurées à leur valeur d'origine.	Idem au résultat attendu.	
4	Vérifier que le contrôle des champs s'effectue correctement pour chaque champ du formulaire.	Une erreur doit être affichée si un champ est vide ou invalide, indiquant quel est le champ concerné.	Idem au résultat attendu.	

5	Supprimer un système d'exploitation qu'un ou plusieurs appareils mobiles utilisent.	Une erreur doit être affichée indiquant le nom des appareils mobiles qui doivent être supprimés avant de pouvoir supprimer le système d'exploitation.	Idem au résultat attendu.	
6	Supprimer un système d'exploitation qu'aucun appareil mobile n'utilise.	Le système d'exploitation doit être supprimé.	Idem au résultat attendu.	

19.1.5.7 Tests de la gestion des applications

Tous les tests suivants s'effectuent lorsque l'administrateur est authentifié.

No	Test effectué	Résultat attendu	Résultat observé	Remarque
1	Créer une nouvelle application ayant le nom d'une application déjà existante.	Un message d'erreur doit être affiché indiquant qu'une application existe déjà sous ce nom-là.	Idem au résultat attendu.	
2	Créer une nouvelle application avec un nouveau nom.	L'application doit être créée.	Idem au résultat attendu.	
3	Choisir de modifier l'application puis modifier certains champs et cliquer sur « cancel modifications ».	Les valeurs des champs doivent être restaurées à leur valeur d'origine.	Idem au résultat attendu.	
4	Vérifier que le contrôle des champs s'effectue correctement pour chaque champ du formulaire.	Une erreur doit être affichée si un champ est vide ou invalide, indiquant quel est le champ concerné.	Idem au résultat attendu.	
5	Supprimer une application.	L'application doit être supprimée ainsi que les liens entre l'application concernée et les fichiers qu'elle contenait. Les fichiers ne sont pas supprimés.	Idem au résultat attendu.	

19.1.5.8 Tests de la gestion des fichiers

Tous les tests suivants s'effectuent lorsque l'administrateur est authentifié.

No	Test effectué	Résultat attendu	Résultat observé	Remarque
1	Ajouter un fichier dont le nom est égal au nom d'un fichier déjà présent sur le serveur.	Un message d'erreur doit être affiché indiquant qu'un fichier existe déjà sous ce nom-là.	Idem au résultat attendu.	
2	Ajouter un nouveau fichier dont le nom n'est pas encore utilisé.	Le fichier doit être ajouté au serveur de fichiers et ses propriétés (taille, code haché, etc.) doivent être automatiquement « calculées » puis ajoutées au serveur de base de données.	Idem au résultat attendu.	
3	Ajouter un fichier de petite taille sur le serveur.	Le fichier doit être envoyé par le navigateur de manière correcte et doit être reçu par le serveur correctement également.	Erreur : le fichier reçu par le serveur est vide. Après de plus amples essais, il s'avère que cette erreur se produit uniquement avec des fichiers de petite taille (~40 octets et moins)	Après des tests divers, l'origine du problème semble être causée au niveau de la librairie d'Apache s'occupant de la gestion de l'upload des fichiers. Le fichier est reçu,

			et que sur le navigateur Firefox et le navigateur Safari. Cette erreur ne se produit pas si SSL n'est pas utilisé. Enfin, cette erreur n'apparaît jamais avec Internet Explorer, qu'il y ait SSL ou non.	mais celui-ci fait une taille de 0 et est totalement vide. Il se peut aussi que l'erreur soit due au serveur Apache lui-même, au moment de la réception de la requête, mais l'erreur est difficilement repérable à ce niveau. Dans tous les cas, cette erreur n'est malheureusement pas corrigible au niveau Java.
4	Choisir de modifier le fichier puis modifier certains champs et cliquer sur « cancel modifications ».	Les valeurs des champs doivent être restaurées à leur valeur d'origine.	Idem au résultat attendu.	
5	Supprimer un fichier.	Le fichier doit être supprimé du serveur de fichiers et ses informations doivent être supprimées du serveur de fichiers. Les liens qu'avait ce fichier avec les applications doivent également être supprimés.	Idem au résultat attendu.	
6	Sélectionner la tâche « modifier un fichier », puis sélectionner un fichier portant un nom différent du fichier original et envoyer ce fichier.	Un message d'erreur doit s'afficher indiquant que le fichier envoyé n'a pas le même nom que le fichier original.	Idem au résultat attendu.	Seul un fichier portant le même nom que le fichier original peut être envoyé au serveur.

19.1.5.9 Tests de la gestion des attributions des fichiers dans les applications

Tous les tests suivants s'effectuent lorsque l'administrateur est authentifié.

No	Test effectué	Résultat attendu	Résultat observé	Remarque
1	Sélectionner une application et essayer de lui ajouter et de lui supprimer des fichiers.	Les fichiers doivent être ajoutés et supprimés de l'application en fonction du choix de l'utilisateur.	Idem au résultat attendu.	Si aucun fichier n'est sélectionné et que l'utilisateur décide d'ajouter ou de supprimer un fichier, un message d'erreur est affiché.

19.1.5.10 Tests de la gestion des attributions des applications dans les groupes

Tous les tests suivants s'effectuent lorsque l'administrateur est authentifié.

No	Test effectué	Résultat attendu	Résultat observé	Remarque
1	Sélectionner un groupe et essayer de lui ajouter et de lui supprimer des applications.	Les applications doivent être ajoutées et supprimées du groupe en fonction du choix de l'utilisateur.	Idem au résultat attendu.	Si aucune application n'est sélectionnée et que l'utilisateur décide d'ajouter ou de supprimer une application, un message d'erreur est affiché.

19.1.5.11 Tests de la gestion des paramètres de sécurité

Tous les tests suivants s'effectuent lorsque l'administrateur est authentifié.

No	Test effectué	Résultat attendu	Résultat observé	Remarque
1	Créer un nouvel ensemble de paramètres de sécurité portant le nom d'un ensemble déjà existant sur le serveur.	Un message d'erreur doit être affiché indiquant qu'un ensemble de paramètres de sécurité avec ce nom existe déjà.	Idem au résultat attendu.	
2	Créer un nouvel ensemble de paramètres de sécurité avec un nouveau nom.	Le nouvel ensemble de paramètres de sécurité doit être créé.	Idem au résultat attendu.	
3	Choisir de modifier un ensemble de paramètres de sécurité puis modifier certains champs et cliquer sur « cancel modifications ».	Les valeurs des champs doivent être restaurées à leur valeur d'origine.	Idem au résultat attendu.	
4	Vérifier que le contrôle des champs s'effectue correctement pour chaque champ du formulaire.	Une erreur doit être affichée si un champ est vide ou invalide, indiquant quel est le champ concerné.	Idem au résultat attendu.	
5	Supprimer un ensemble de paramètres de sécurité qu'un ou plusieurs groupes utilisent.	Une erreur doit être affichée indiquant le nom des groupes qui doivent être supprimés avant de pouvoir supprimer l'ensemble des paramètres de sécurité.	Idem au résultat attendu.	
6	Supprimer un ensemble de paramètres de sécurité qu'aucun groupe n'utilise.	L'ensemble des paramètres de sécurité doit être supprimé.	Idem au résultat attendu.	

19.1.5.12 Tests de la gestion des paramètres du serveur

Tous les tests suivants s'effectuent lorsque l'administrateur est authentifié.

No	Test effectué	Résultat attendu	Résultat observé	Remarque
1	Choisir de modifier les paramètres du serveur puis modifier certains champs et cliquer sur « cancel modifications ».	Les valeurs des champs doivent être restaurées à leur valeur d'origine.	Idem au résultat attendu.	La modification des paramètres du serveur nécessite que celui-ci soit redémarré afin qu'il prenne en compte les nouveaux paramètres.
4	Vérifier que le contrôle des champs s'effectue correctement pour chaque champ du formulaire.	Une erreur doit être affichée si un champ est vide ou invalide, indiquant quel est le champ concerné.	Idem au résultat attendu.	

19.1.6 Tests d'ordre généraux

No	Test effectué	Résultat attendu	Résultat observé	Remarque
1	Se connecter au service avec deux appareils mobiles simultanément en effectuant des tâches diverses.	Le client lourd doit s'exécuter sans générer d'erreurs inattendues.	Aucun problème n'a été détecté lors de ce test, ce qui confirme le bon fonctionnement du serveur.	

2	Effectuer le test 1, mais en se connectant avec le même nom d'utilisateur sur les deux appareils mobiles.	Aucune restriction n'est existante quant au nombre de sessions ouvertes simultanément sur un serveur par un même utilisateur sur différents appareils mobiles. L'exécution doit donc se dérouler normalement.	Aucun problème n'a été détecté.	
3	Se connecter au service à l'aide d'un appareil mobile, puis essayer de se déplacer entre des salles.	Aucune perte de connexion ne doit se produire et le client lourd doit fonctionner de manière normale.	Aucun problème n'a été détecté.	
4	Durant le test 3, démarrer quelques simulations d'applications et laisser le programme s'exécuter durant un temps un peu plus prolongé (>2 heures).	Les requêtes simulées doivent s'effectuer correctement et les authentifications doivent être demandées en fonction des critères établis dans les paramètres de sécurité.	Aucun problème n'a été détecté.	
5	Mettre à jour un fichier sur le serveur lorsque celui-ci est en cours de téléchargement par un client.	La modification du fichier sur le serveur ne doit pas être exécutée pendant que le fichier est en cours de transfert sur le client. La mise à jour du fichier doit donc être mise en attente jusqu'à ce que le transfert soit terminé.	Idem au résultat attendu.	La gestion des accès aux fichiers est gérée par le serveur WebDAV lui-même.
6	Déconnecter le serveur ou mettre le PDA en veille lors du transfert d'une application.	Le transfert doit être interrompu et une erreur doit être affichée.	Erreur : le transfert est interrompu, mais aucune erreur n'est affichée. L'application semble être figée, bien que l'interface graphique réagisse encore.	Lors d'une telle interruption, aucune exception n'est levée au niveau Java et la lecture du flux de données se bloque. Cela est dû à un problème de détection de l'interruption par Java, par la machine virtuelle ou par le système d'exploitation sous-jacent. Il n'est ainsi pas possible de corriger ce problème au niveau du code, à moins d'implémenter un contrôle du transfert à l'aide de threads, ce qui alourdirait considérablement l'application. Il est à noter que dans les autres cas, les problèmes sont correctement détectés.

19.2 Manuel d'utilisation du client lourd

19.2.1 Démarrage du client lourd

Pour démarrer le client lourd, ouvrir l'explorateur de fichiers de Windows Mobile et ouvrir le répertoire dans lequel se trouve le client lourd. Cliquer ensuite sur le raccourci de démarrage de l'application, nommé *start* dans ce cas.

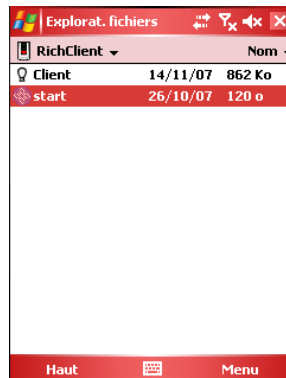


Figure 33: Démarrage du client lourd

19.2.2 Page d'accueil

La page d'accueil permet de vérifier le nom et la version du client lourd. Cette page est affichée pendant 3 secondes avant d'être fermée et remplacée par la fenêtre de gestion des mises à jour.

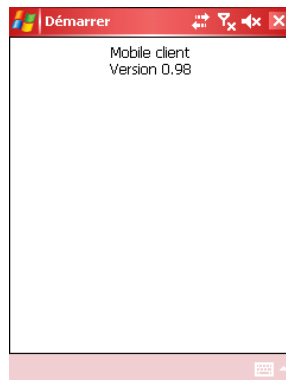


Figure 34: Page d'accueil

19.2.3 Mise à jour du client lourd

Le client lourd vérifie à chaque démarrage si sa version est valide par rapport à la version stockée sur le serveur. Si tel n'est pas le cas, le client lourd télécharge la nouvelle version, puis affiche un message indiquant qu'un redémarrage est nécessaire à la fin de l'installation. Il suffit alors de cliquer sur le bouton « ok », puis d'attendre la fin de l'installation. Un message s'affiche dans la console lorsque l'installation est terminée et que l'utilisateur peut fermer et ouvrir à nouveau l'application.

Les figures 35, 36 et 37 représentent les différentes étapes lors de la mise à jour du client lourd.

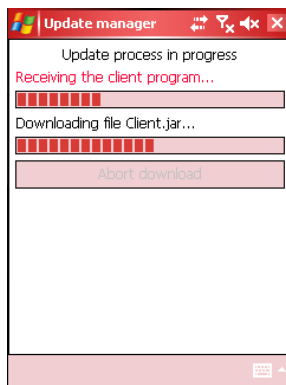


Figure 35: Mise à jour du client lourd

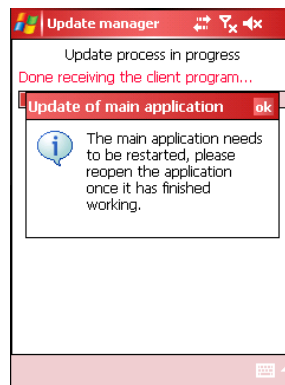


Figure 37: Avertissement de redémarrage du client lourd

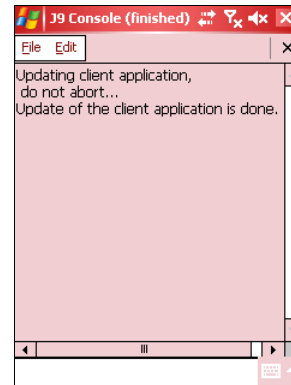


Figure 36: Fin de la mise à jour du client lourd

19.2.4 Mise à jour de la librairie d'authentification

Une fois le client lourd à jour, celui-ci va vérifier l'intégrité de sa librairie d'authentification. Si la librairie est obsolète ou invalide, le client lourd télécharge la nouvelle version sur le serveur et l'installe. Aucun redémarrage n'est nécessaire dans ce cas-là.

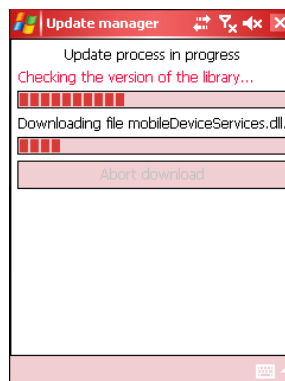


Figure 38: Mise à jour de la librairie d'authentification

19.2.5 Authentification matérielle

Le client lourd et la librairie d'authentification étant à jour, le client déclenche une authentification matérielle afin d'identifier l'appareil mobile utilisé. Aucune action n'est requise de la part de l'utilisateur.

Il est à noter que la tentative de fermeture de la fenêtre d'une quelconque authentification engendre la fermeture du client lourd dans le but d'éviter tout risque de sécurité.

Cette authentification est également déclenchée lors des contrôles périodiques longs.

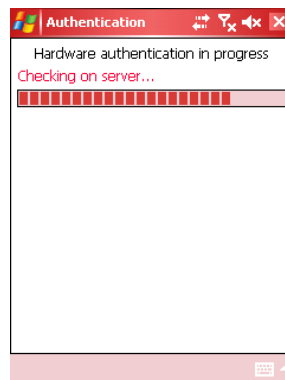


Figure 39:
Authentification
matérielle

19.2.6 Authentification formelle

L'authentification formelle suit l'authentification matérielle. La fenêtre est composée d'une liste contenant les noms des utilisateurs autorisés à utiliser l'appareil mobile. Lors des authentifications formelles ultérieures, seul le nom de l'utilisateur qui s'est authentifié est proposé.

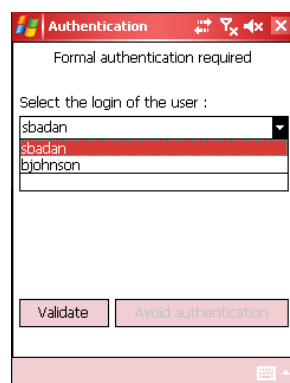


Figure 40: Liste de choix
des utilisateurs

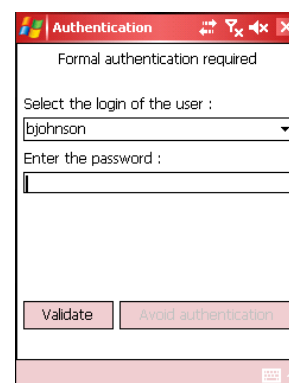


Figure 41: Fenêtre
d'authentification
formelle

Deux boutons sont affichés dans la fenêtre. Le bouton *Validate* qui permet de valider l'authentification et le bouton *Avoid authentication* qui permet, lorsque le bouton est activé, d'éviter temporairement l'authentification formelle. Dans ce cas, l'authentification formelle est remplacée par une authentification externe et une nouvelle authentification formelle est requise une fois le délai d'évitement échu.

19.2.7 Authentification externe

L'authentification externe est effectuée périodiquement ou lorsqu'une application est démarrée en dehors du délai de démarrage des applications. Cette authentification effectue un balayage des appareils Bluetooth présents afin de contrôler que le badge ou le téléphone Bluetooth servant d'identificateur soit toujours présent. Si tel n'est pas le cas, l'authentification est considérée comme invalide. Cette authentification ne requiert donc aucune interaction avec l'utilisateur.

Si le dispositif Bluetooth est désactivé, une erreur est affichée et le client lourd est automatiquement fermé après la fermeture de l'erreur.

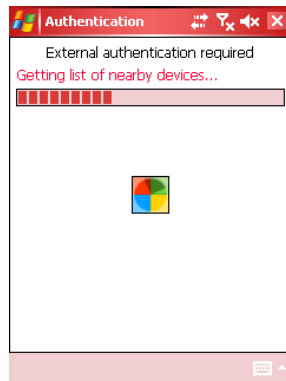


Figure 43:
Authentification externe
par Bluetooth

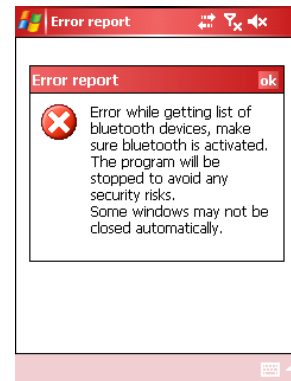


Figure 42: Erreur de
Bluetooth lors de
l'authentification externe

19.2.8 Mise à jour des applications

Le service de mise à jour procède en deux étapes.

Durant la première étape, le service de mise à jour vérifie que toutes les applications prioritaires soient installées et valides, et si tel n'est pas le cas, il télécharge les mises à jour ou les installations nécessaires sans laisser de choix à l'utilisateur.

Dans la deuxième étape, le service de mise à jour vérifie quelles sont les applications installées et vérifie également leur intégrité. Si au moins une mise à jour a été détectée ou si l'un des fichiers est invalide, une fenêtre de choix telle que représentée dans la figure 44 s'ouvre :

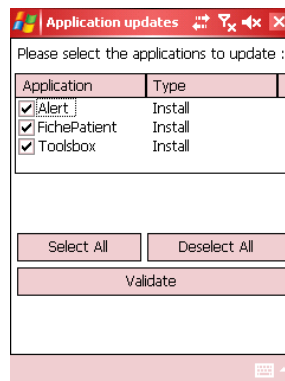


Figure 44: Fenêtre de
choix des mises à jour

Cette fenêtre est composée d'une liste des applications et du type d'action à effectuer sur celles-ci :

- ◇ Install : l'application concernée n'est pas installée ou certains fichiers sont manquants. Si cette application est sélectionnée, les fichiers manquants ou invalides seront téléchargés. Dans le cas contraire, tous les fichiers partiellement installés de cette application sont supprimés, à moins que l'un de ces fichiers soit utilisé par une autre application.
- ◇ Update : l'application concernée possède tous ses fichiers, mais certains de ceux-ci sont

invalides ou ne sont pas à jour. Si l'application est installée, les fichiers nécessitant une mise à jour ou une correction sont à nouveau téléchargés. Dans le cas contraire, aucune action n'est effectuée sur ces fichiers.

- ◇ **Mandatory** : l'application concernée possède tous ses fichiers, mais certains de ceux-ci sont invalides ou ne sont pas à jour. La mise à jour de cette application est obligatoire et ne peut pas être évitée.

Cette fenêtre comporte par ailleurs trois boutons. Le bouton *Select All* permet de sélectionner toutes les mises à jour. Le bouton *Deselect All* permet de désélectionner toutes les applications dont la mise à jour est de type *Install* ou *Update*. Enfin, le bouton *Validate* permet de valider le choix.

19.2.8.1 Transfert des fichiers

Lors du transfert d'une application sur le client mobile, une barre d'état du téléchargement est affichée, comme le représente la figure 45.

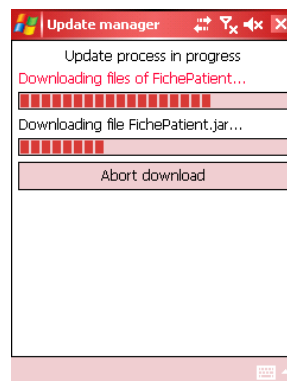


Figure 45: Transfert d'une application non prioritaire

Le transfert d'applications non prioritaires, qu'elles soient de types *Mandatory* ou *Update*, peut être annulé. L'annulation d'un transfert engendre la suppression du fichier en cours de transfert ainsi que de tous les fichiers contenus dans l'application dont le transfert a été annulé. Les fichiers qui sont utilisés par d'autres applications ne sont en revanche pas supprimés.

Pour annuler le transfert d'une application, il suffit de cliquer sur le bouton *Abort download*. Un message d'avertissement est alors affiché à l'écran :

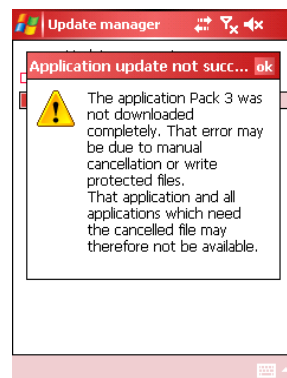


Figure 46: Annulation d'un téléchargement

19.2.8.2 Contrôle des liaisons entre les fichiers des applications

Plusieurs applications peuvent utiliser un même fichier, appelé fichier commun. Lors de la mise à jour d'une application possédant un tel fichier, le fichier commun est également mis à jour. Il se peut alors que les autres applications utilisant ce fichier, si elles ne sont pas mises à jour, génèrent des erreurs de compatibilité de version lors de l'utilisation du nouveau fichier commun.

Dans ce but, une vérification s'exécutant en fin de processus de mise à jour contrôle si des fichiers d'applications non mises à jour ont été modifiés par d'autres applications. Si tel est le cas, un message d'avertissement est affiché et les applications concernées sont mises à jour automatiquement. L'utilisateur peut cependant annuler la mise à jour s'il le souhaite, engendrant la suppression de l'application.

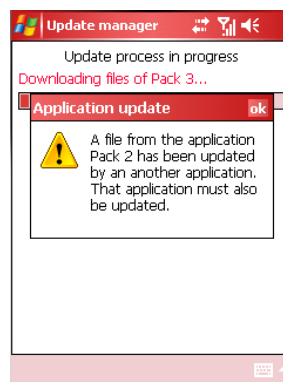


Figure 47: Contrôle des fichiers communs à plusieurs applications

19.2.8.3 Application de test

Une application de test a été implémentée afin de s'assurer du bon fonctionnement du client lourd. Cette application se présente de la manière suivante :

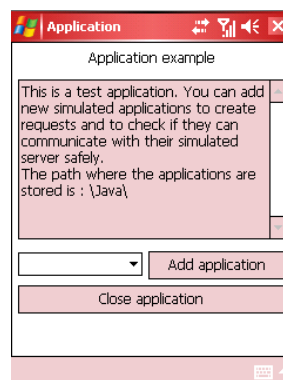


Figure 48: Application de test

Cette application permet de simuler l'exécution simultanée de plusieurs applications. Une application en exécution va effectuer périodiquement une demande d'identifiant de sécurité d'applications au client de sécurité (la période des demandes étant calculée aléatoirement entre 1 et 10 secondes). Le client de sécurité vérifie alors si l'utilisateur est autorisé à utiliser l'application et retourne l'identifiant si tel est le cas. Dans ce cas, l'application simulée envoie cet identifiant au serveur de sécurité afin d'obtenir le nom de l'utilisateur qui utilise cette application, le nom étant alors affiché à l'écran. Dans le cas contraire, un

message indique simplement que l'identifiant est invalide.

La figure 49 représente plusieurs applications en cours d'exécution. Il est possible d'observer le résultat de deux applications différentes : *ToolsBox.jar* et *abcd*. La première est une application installée et autorisée à être exécutée par l'utilisateur, tandis que le nom de la seconde a été saisi dans le simulateur d'applications et n'est pas autorisée à être exécutée. Il est possible d'observer que la première application retourne le nom de l'utilisateur, ce qui indique que l'identifiant de sécurité est valide. En revanche, la seconde application ne reçoit pas d'identifiant valide, c'est pourquoi le message « check invalid » est affiché.

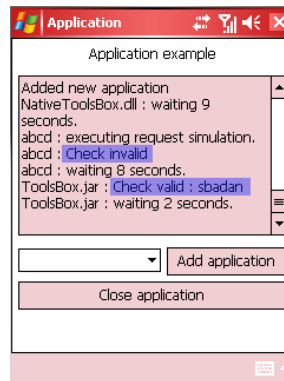


Figure 49: Résultats des requêtes simulées

Remarque :

En pratique, cette application sera remplacée par le chargeur d'applications implémenté dans le cadre d'un autre travail de diplôme. Il faut par ailleurs noter que l'application n'enverrait pas directement de requête au serveur de sécurité dans le but de vérifier l'identifiant de sécurité des applications. Cette requête serait en effet générée par le serveur de l'application, lors de la réception de la requête de la part de l'application.

Pour ajouter une application simulée, deux solutions sont possibles. La première solution consiste à sélectionner l'un des fichiers installés dans la liste déroulante, puis de cliquer sur « Add application ». La seconde possibilité est de saisir le nom de l'application dans la liste, puis de cliquer sur « Add application ». La première possibilité est très utile lorsque l'on souhaite vérifier que les applications installées sont autorisées à être utilisées, tandis que la seconde possibilité permet de tester des applications non autorisées afin de s'assurer que le client de sécurité les refuse.

19.2.9 Messages d'erreur

Le client lourd est une application rigide laissant une très faible tolérance aux erreurs. Ainsi, lorsque des anomalies sont détectées au niveau du client ou au niveau du serveur, des messages d'erreur sont affichés à l'écran et le client lourd est généralement fermé automatiquement. En fonction de la gravité de l'erreur détectée, la fermeture peut se faire de manière habituelle en laissant le temps à chaque application de se fermer ou de manière brusque en fermant la machine virtuelle elle-même afin de minimiser les risques de sécurité.

Deux messages d'erreurs susceptibles d'être rencontrés en utilisation normale par l'utilisateur sont présentés ci-dessous.

19.2.9.1 Identifiant de sécurité invalide

Lorsqu'aucune authentification formelle n'est plus effectuée durant un délai prolongé (environ 1 jour, en fonction du délai de validité des sessions paramétré par l'administrateur¹⁷), un message d'erreur tel que celui-ci risque de s'afficher lors de l'authentification :

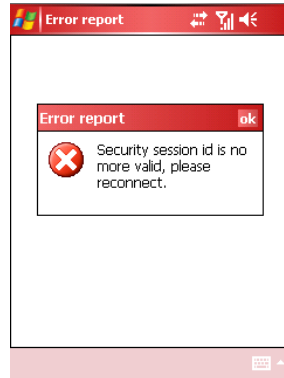


Figure 50: Identifiant de sécurité invalide

Cette erreur est due à l'expiration de la session de l'utilisateur. En effet, chaque session ouverte sur le serveur possède un certain délai paramétré par l'administrateur du service. Si ce délai est échu, la session est supprimée du serveur. Dans ce cas, si une nouvelle demande d'authentification provenant du client lourd est reçue par le serveur, celle-ci sera refusée, car la session de l'utilisateur n'existe plus. Cette erreur peut aussi provenir d'un redémarrage du serveur. Dans un tel cas, le client lourd est fermé et l'utilisateur est prié de se reconnecter.

19.2.9.2 Compte de l'utilisateur ou de l'appareil mobile bloqué

Lorsqu'une authentification échoue plusieurs fois, le compte de l'utilisateur est bloqué. Le nombre de tentatives dépend des paramètres choisis par l'administrateur du service. L'administrateur peut aussi bloquer le compte de l'utilisateur à distance s'il le souhaite, pour des raisons de sécurité¹⁸. Lorsque le compte de l'utilisateur est bloqué, le message suivant est affiché :

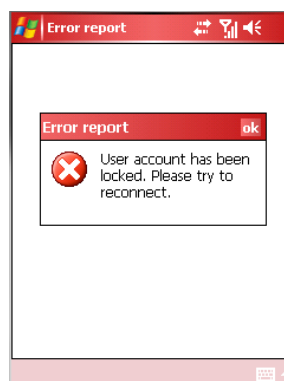


Figure 51: Compte de l'utilisateur bloqué

Dans ce cas-là, le client lourd est fermé et l'utilisateur est prié de se rendre chez l'administrateur du service pour débloquer le compte. L'erreur est similaire pour le blocage du compte de l'appareil mobile.

¹⁷ Pour obtenir plus d'informations concernant la modification du délai de validité des sessions ouvertes, voir le chapitre 19.3.11, page 107.

¹⁸ Pour obtenir plus d'informations concernant la désactivation d'un compte d'utilisateur, voir le chapitre 19.3.3, page 101.

19.3 Manuel d'utilisation du service d'administration

Le service d'administration permet à l'administrateur de gérer la totalité des paramètres du serveur de sécurité. Les détails de ces fonctionnalités sont donnés ci-dessous.

19.3.1 Authentification

Pour s'authentifier sur le site Internet d'administration, il faut utiliser le formulaire d'authentification à droite de la page, saisir ensuite le nom de l'administrateur autorisé à se connecter ainsi que son mot de passe, puis cliquer sur *Sign In*. Pour se déconnecter, cliquer sur *Log Out*.

Authentication

Administrator ID:

Password:

Figure 52: Connexion sur le site d'administration

19.3.2 Sélection des tâches

Une fois connecté, le menu de droite (représenté en vert dans la figure 53) permet de sélectionner la tâche désirée. Les tâches sont réparties en catégories dans le but de faciliter l'accès à celles-ci. Ce menu est affiché peut importe la tâche sélectionnée, à l'exception du cas où l'administrateur n'est pas authentifié.

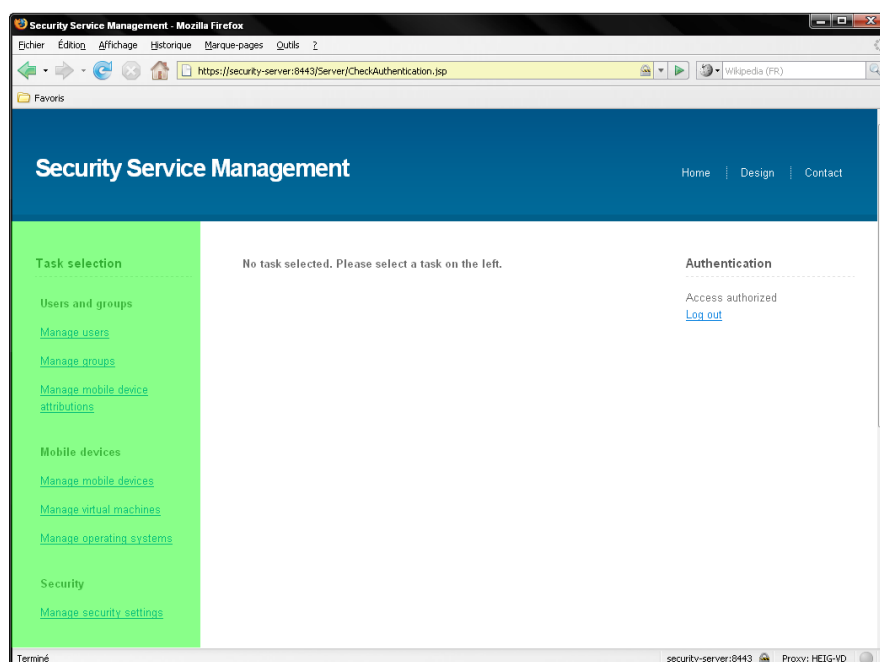


Figure 53: Représentation du menu des tâches

Lorsqu'une tâche est sélectionnée, la partie centrale du site affiche ce que l'on souhaite effectuer avec la tâche. Par exemple, en cliquant sur « Manage Users », la partie centrale propose les éléments

présentés dans la figure 54.

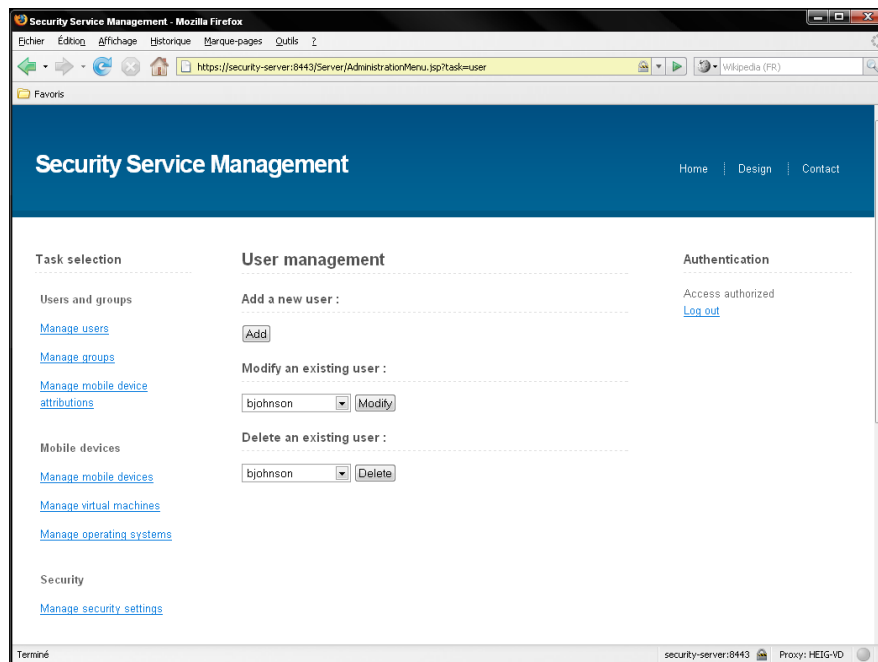


Figure 54: Gestion des utilisateurs

Généralement, une tâche permet d'ajouter, modifier ou supprimer un élément choisi, ici des utilisateurs. Certaines tâches peuvent cependant proposer des actions spécifiques comme l'attribution d'appareils mobiles par exemple, où le choix est alors restreint à la modification seulement.

19.3.3 Gestion des utilisateurs – Manage users

Il est possible d'ajouter, de modifier ou de supprimer des utilisateurs du service de sécurité. Les paramètres de l'utilisateur sont les suivants :

- ◇ Le nom de login de l'utilisateur. Ce nom ne doit être composé que de lettres ou de chiffres. Les signes ponctués ne sont pas permis. Ce nom est utilisé lors des authentifications des utilisateurs sur les clients mobiles, c'est pourquoi il doit être unique.
- ◇ Le nom et le prénom de l'utilisateur;
- ◇ Le groupe auquel l'utilisateur appartient. Un utilisateur ne peut appartenir qu'à un seul groupe à la fois.
- ◇ Le mot de passe formel de l'utilisateur qui sera demandé lors des authentifications formelles. Ce mot de passe est haché afin de ne plus être lisible une fois l'utilisateur ajouté ou modifié, il est donc tout à fait normal de ne pas pouvoir le lire lors d'une modification ultérieure de l'utilisateur.
- ◇ Le nombre d'authentifications invalides formelles et externes que l'utilisateur a effectuées; Ces valeurs indiquent le nombre de fois que l'utilisateur s'est authentifié de manière invalide pour chacun des types d'authentification.
- ◇ La clé d'authentification externe : cette valeur correspond ici à l'adresse MAC (*Medium Access Control*) de l'appareil mobile Bluetooth porté sur l'utilisateur et servant d'authentifiant externe, tel qu'un badge ou téléphone mobile.

- ◇ L'état du compte de l'utilisateur. En cochant la case, le compte de l'utilisateur est désactivé et l'utilisateur ne pourra plus s'authentifier. En décochant la case, le compte de l'utilisateur est à nouveau activé et les compteurs d'authentications invalides sont remis à zéro.

Un nouvel utilisateur n'a par défaut aucune autorisation d'utilisation d'appareils mobiles. Il faut lui spécifier quels sont les appareils mobiles qu'il a le droit d'utiliser. Pour ce faire, suivre la procédure citée dans la section 19.3.7.

Remarque :

L'ajout d'un utilisateur ne peut se faire que si au moins un groupe a été préalablement créé.

19.3.4 Gestion des groupes – Manage groups

Un groupe est défini par les éléments suivants :

- ◇ Un nom qui doit être unique;
- ◇ Les paramètres de sécurité appliqués au groupe et à tous les utilisateurs qu'il contient;

Un groupe ne peut suivre qu'un seul ensemble de paramètres de sécurité (*security settings*), mais peut être autorisé à utiliser zéro, une ou plusieurs applications.

Pour spécifier quelles sont les applications que le groupe est autorisé à utiliser, suivre la procédure décrite à la section 19.3.10.4.

Remarque :

L'ajout d'un groupe ne peut se faire que si au moins un ensemble de paramètres de sécurité a été préalablement créé.

19.3.5 Gestion des paramètres de sécurité – Manage security settings

Les paramètres de sécurité (nommés *security settings* dans le cadre du programme) définissent les règles d'authentification et les règles de sécurité des clients lourds. Ces règles contiennent :

- ◇ Un nom permettant de reconnaître l'ensemble de règles;
- ◇ Un nombre qui définit le nombre maximum d'authentications invalides formelles avant que le compte de l'utilisateur ne soit bloqué. Ce nombre doit être supérieur à 1.
- ◇ Un nombre qui définit le nombre maximum d'authentications invalides externes avant que le compte de l'utilisateur ne soit bloqué. Ce nombre doit être supérieur à 1.
- ◇ Le délai d'évitement qui définit le délai après lequel une nouvelle authentification formelle est demandée à la suite d'un évitement d'authentification formelle. Ce délai est donné en secondes et doit être supérieur ou égal à 10 secondes.
- ◇ L'état de la possibilité d'éviter les authentifications, possibilité qui peut être activée ou désactivée; si l'évitement est désactivé, l'utilisateur ne pourra plus éviter les authentifications formelles. Dans le cas contraire, l'authentification formelle peut être contournée temporairement, engendrant une authentification externe. Une nouvelle authentification formelle est ensuite demandée après échéance du délai d'évitement.
- ◇ Le délai de démarrage des applications après lequel le démarrage d'une nouvelle application déclenche une demande d'authentification; ce délai est donné en secondes et doit être

supérieur à 0.

- ◇ Le délai d'authentification court périodique entre deux authentifications externes; ce délai est donné en secondes et doit être supérieur à 30 secondes.
- ◇ Le délai d'authentification long périodique entre deux authentifications matérielles et formelles; ce délai est donné en secondes et doit être à la fois supérieur à 30 et à la fois supérieur au délai d'authentification court périodique.
- ◇ Le délai de transition des mises à jour après lequel une mise à jour optionnelle devient obligatoire; ce délai est donné en heures et doit être supérieur à 0.
- ◇ L'état de la possibilité de définir un délai de transition, possibilité qui peut être activée ou désactivée; si le délai de transition est désactivé, toutes les mises à jour deviennent obligatoires. Dans le cas contraire, le serveur se base sur ce délai de transition pour décider si la mise à jour de l'application est obligatoire ou non.

19.3.6 Gestion des appareils mobiles – Manage mobile devices

Chaque appareil mobile désirant se connecter au service doit préalablement être inscrit au niveau du serveur de sécurité, sans quoi l'accès au service pour cet appareil serait refusé.

Il est donc possible d'ajouter, de modifier ou de supprimer un appareil mobile. Un tel appareil possède les propriétés suivantes :

- ◇ Un nom permettant de le reconnaître;
- ◇ Le code IMEI permettant d'identifier l'appareil mobile de manière unique; Ce code peut être trouvé sur l'appareil mobile en tapant le numéro de téléphone **#06#*.
- ◇ Le système d'exploitation utilisé par l'appareil mobile;
- ◇ La disponibilité d'un système d'authentification externe sur l'appareil mobile (disponible ou non); Dans ce cas-ci, l'authentification externe correspond à une authentification par la technologie Bluetooth. Si l'appareil mobile dispose d'un module Bluetooth, la case doit être cochée. Dans le cas contraire, la case doit être décochée afin que les authentifications externes soient remplacées par des authentifications formelles et que l'évitement des authentifications formelles ne soit plus possible.
- ◇ L'état du compte de l'utilisateur (activé ou désactivé); si le compte est désactivé, l'utilisateur n'est plus autorisé à s'authentifier. Si l'utilisateur est en train d'utiliser le service au moment de la désactivation du compte, il est automatiquement déconnecté à la prochaine authentification.

Remarque :

L'ajout d'un appareil mobile ne peut se faire que si au moins un système d'exploitation a été préalablement créé.

19.3.7 Gestion des attributions des appareils mobiles – Manage mobile device attributions

Les appareils mobiles qu'un utilisateur a le droit d'utiliser doivent être paramétrés ici. Par défaut, un nouvel utilisateur n'a le droit d'utiliser aucun appareil mobile.

Il est à noter que plusieurs utilisateurs peuvent avoir le droit d'utiliser un même appareil mobile. Le client lourd se base sur ce lien entre les utilisateurs et les appareils mobiles afin de proposer, au moment de la première authentification formelle sur le client lourd, le nom des utilisateurs ayant le

droit de se connecter sur l'appareil mobile.

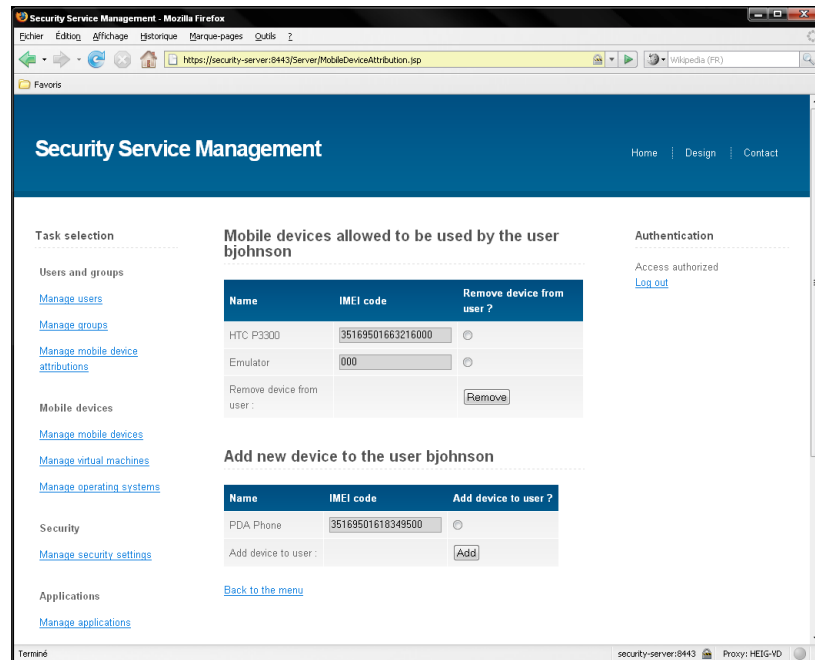


Figure 55: Attribution des appareils mobiles

Pour ajouter un appareil mobile dans la liste des appareils que l'utilisateur est autorisé à utiliser, sélectionner un appareil mobile et cliquer sur le bouton « add ».

Pour enlever un appareil mobile de la liste des appareils que l'utilisateur est autorisé à utiliser, sélectionner un appareil mobile et cliquer sur le bouton « remove ».

19.3.8 Gestion des machines virtuelles – Manage virtual machines

Il est nécessaire de définir quelles sont les machines virtuelles des clients mobiles autorisées à être utilisées sur les clients lourds. Une machine virtuelle est définie par :

- ◇ Le nom de la machine virtuelle;
- ◇ La version de la machine virtuelle;

Ces deux paramètres définissent la machine virtuelle de manière unique. Il est à noter que la machine virtuelle utilisée d'IBM possède les propriétés suivantes :

- ◇ Nom : J9
- ◇ Version : 2.3

19.3.9 Gestion des systèmes d'exploitation – Manage operating systems

De manière similaire aux machines virtuelles, il est nécessaire de définir quels sont les systèmes d'exploitation autorisés à être utilisés au niveau des appareils mobiles. Un système d'exploitation est défini par :

- ◇ Un nom, une version et une architecture;
- ◇ Le nom du programme de client lourd que le système d'exploitation doit utiliser. Ce paramètre permet, si plusieurs versions du client lourd existent pour des versions de systèmes

d'exploitation différentes, de spécifier quelle version du client lourd il faut utiliser pour un système d'exploitation donné.

- ◇ Le nom de la machine virtuelle à utiliser pour ce système d'exploitation;

Remarque :

L'ajout d'un système d'exploitation ne peut se faire que si au moins une machine virtuelle a été préalablement créée et si l'application du client lourd a été préalablement ajoutée sur le serveur.

Pour trouver la version du système d'exploitation sous Windows Mobile, aller dans les menus *Démarrer\Paramètres\Systèmes* et sélectionner *A propos de*. Le numéro de version est alors donné sous la forme *Microsoft Windows Mobile Version 5.0 OS 5.1.195 (Build ...)*. Le numéro à retenir est 5.1.195. A partir de ce numéro, le système d'exploitation peut être représenté de la manière suivante dans le service d'administration :

- ◇ Nom : Windows CE
- ◇ Version : 5.1 build 195
- ◇ Architecture : arm

19.3.10 Gestion des applications et des fichiers

L'ajout d'une application sur le serveur se déroule en quatre étapes. La première étape consiste à créer l'application, à lui donner un nom et à définir ses propriétés d'installation. Une fois cette étape réalisée, il faut ajouter les fichiers de l'application sur le serveur. Ensuite, il faut spécifier, pour cette application, quels sont les fichiers qu'elle contient. Enfin, il est nécessaire de spécifier quels sont les groupes autorisés à utiliser la nouvelle application. Les détails de chacune de ces étapes sont donnés ci-dessous.

19.3.10.1 Gestion des applications – Manage applications

Une application, comme définie dans le chapitre 12.1, est définie par les propriétés suivantes :

- ◇ Un nom d'application unique;
- ◇ Un état définissant si l'application est prioritaire ou non; si l'application est définie comme étant prioritaire, l'installation de cette application se fera obligatoirement, à moins que l'application ne soit pas installable automatiquement, auquel cas une erreur serait affichée et le client lourd serait alors automatiquement fermé.
- ◇ Un état définissant si l'application est installable automatiquement ou non, ce qui est utile dans le cas où l'installation de l'application requiert des opérations spéciales ou si elle est trop volumineuse pour être téléchargée par les appareils mobiles; si l'application est définie comme étant installable automatiquement, le client lourd télécharge tous les fichiers contenus dans cette application et les installe. Dans le cas contraire, une erreur est affichée si l'application est prioritaire et l'application n'est pas installée et/ou supprimée si l'application est non prioritaire, en fonction de la catégorie à laquelle elle appartient.

Une fois une application créée, il est possible de spécifier les fichiers que contient cette application.

19.3.10.2 Gestion des fichiers – Manage files

Cette tâche permet d'ajouter, de modifier ou de supprimer des fichiers sur le serveur.

Un fichier est défini par :

- ◇ Un nom de fichier unique;
- ◇ Le code haché du fichier permettant aux clients de vérifier l'intégrité des fichiers;
- ◇ La date d'ajout du fichier;
- ◇ La taille du fichier;
- ◇ Un état pour savoir si la mise à jour du fichier est prioritaire ou non; Si la mise à jour est définie comme étant prioritaire, le délai de transition des mises à jour des applications est ignoré et tous les fichiers de l'application contenant ce fichier sont mis à jour obligatoirement. Dans le cas contraire, le serveur se base sur le délai de transition défini dans les paramètres de sécurité du groupe de l'utilisateur pour savoir si l'application doit être mise à jour obligatoirement ou non.

Il faut noter que tous ces paramètres sont calculés automatiquement au moment de l'ajout des fichiers sur le serveur, à l'exception de l'état de priorité qui doit être défini par l'administrateur. Le nom du fichier envoyé est utilisé pour identifier le fichier de manière unique.

Un fichier peut aussi être modifié. Il suffit pour cela de choisir la tâche de modification du fichier, puis d'envoyer le nouveau fichier sur le serveur. Il n'est cependant possible de modifier un fichier que par un fichier portant le même nom que le fichier original.

19.3.10.3 Gestion des fichiers contenus dans les applications

Une fois que les nouveaux fichiers sont ajoutés et que de nouvelles applications ont été définies sur le serveur, il est nécessaire définir quels sont les fichiers contenus dans une application donnée. Pour ce faire, il faut sélectionner une application, puis choisir quels sont les fichiers à ajouter ou à supprimer de l'application choisie.

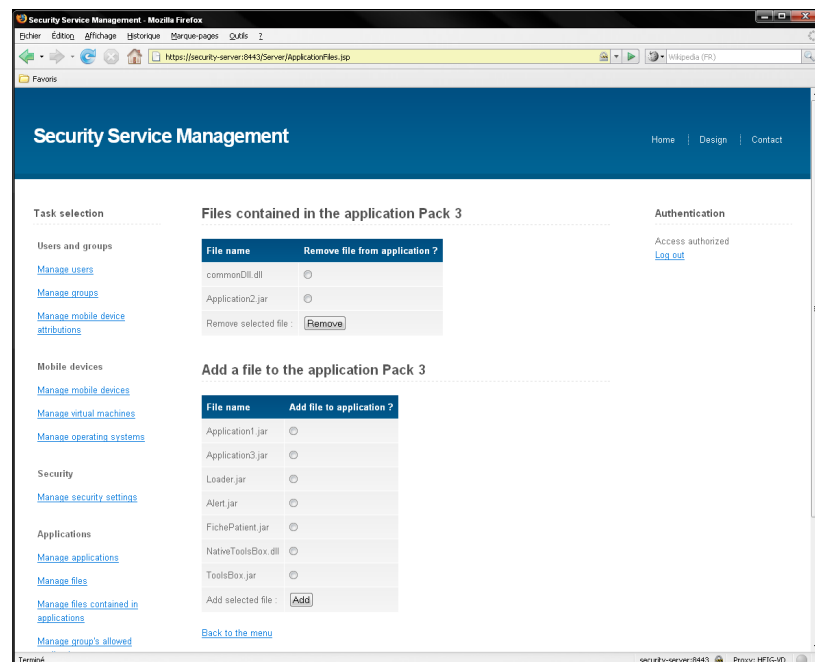


Figure 56: Ajout et suppression de fichiers dans une application

Remarque :

La liste contenant les fichiers qu'il est possible d'ajouter à une application cache les fichiers utilisés par les applications de type « client lourd » (toutes les applications qui sont référées par des systèmes d'exploitation), en raison de la gestion différente des mises à jour de ces fichiers par rapport à des fichiers d'application habituels.

Dans le cas où l'on souhaite ajouter une nouvelle application de client lourd et que l'on souhaite lui attribuer certains fichiers utilisés par d'autres clients lourds, il est nécessaire qu'au moins un système d'exploitation réfère préalablement à cette application afin que la page d'attribution des fichiers affiche aussi fichiers de clients lourds.

19.3.10.4 *Gestion des droits d'exécution des applications - Manage group's allowed applications*

Dans un dernier temps, il est nécessaire de spécifier quels sont les groupes autorisés à utiliser les applications. Il lui suffit pour cela de choisir le groupe d'utilisateurs, puis d'ajouter ou de supprimer les applications que le groupe est autorisé à utiliser.

Remarque :

Les applications de type « client lourd » (applications référées par au moins un système d'exploitation) ne sont pas affichées dans la liste des applications en raison de la gestion différente de ce type d'applications par rapport aux applications habituelles.

19.3.11 *Gestion de la configuration du serveur*

Les paramètres du serveur peuvent également être modifiés depuis le service d'administration. Ces paramètres sont :

- ◇ Le nom de login au service d'administration utilisé pour authentifier l'administrateur;
- ◇ Le mot de passe d'accès au service d'administration utilisé pour authentifier l'administrateur;
- ◇ L'adresse vers le serveur de fichiers;
- ◇ Le nom de login pour la connexion au serveur de fichiers;
- ◇ Le mot de passe pour la connexion au serveur de fichiers;
- ◇ Le répertoire dans lequel les fichiers sont stockés au niveau du serveur de fichiers;
- ◇ Le délai de vérification des sessions invalides. Cette vérification passe en revue toutes les sessions ouvertes par les utilisateurs et supprime celles dont la durée de validité est échue. Ce délai permet de définir la périodicité des vérifications.
- ◇ Délai de validité des sessions. Ce délai spécifie le délai durant lequel une session d'utilisateur est valide. Si ce délai échoit, la session est supprimée à la prochaine vérification des sessions.

Remarque :

La modification de l'un de ces paramètres nécessite un redémarrage du serveur afin qu'il prenne en compte les nouveaux paramètres.

Il existe par ailleurs d'autres paramètres pour le serveur présents dans un fichier de configuration nommé *configuration.xml* et stocké sur le serveur Web dans le répertoire `\webapps\Server\WEB-INF` de ce dernier. Ce fichier contient les paramètres suivants :

- ◇ DBPATH : représente le chemin vers le serveur de gestion de base de données;
- ◇ DBNAME : spécifie le nom de la base de données à utiliser;
- ◇ DBUSER : spécifie le nom de l'utilisateur à utiliser pour se connecter à la base de données;
- ◇ DBPASS : spécifie le mot de passe à utiliser pour se connecter à la base de données;
- ◇ TRUSTSTOREPATH : spécifie le chemin vers le certificat utilisé pour se connecter au serveur de fichiers;
- ◇ TRUSTSTOREPASSWORD : spécifie le mot de passe du certificat susmentionné;
- ◇ PARAMETERSID : spécifie l'identificateur du tuple de la base de données où sont stockés les paramètres de sécurité du serveur;

19.4 Intégration des différentes parties du client lourd

Le client de sécurité étant traité et implémenté dans le cadre de trois travaux de diplôme différents, ce guide a pour but de montrer comment intégrer les différentes parties afin qu'elles puissent fonctionner ensemble.

Au niveau du client lourd, il est nécessaire d'intégrer le chargeur d'applications dans le client de sécurité. Le client de sécurité est composé de trois paquetages différents :

- ◇ Paquetage *client* qui contient le client de sécurité;
- ◇ Paquetage *xmlClasses* qui contient les classes utilisées pour stocker les requêtes XML au format Java;
- ◇ Paquetage *HeavyClient* qui est prévu pour accueillir le chargeur d'applications. Il contient actuellement l'application de simulation de requêtes;

Pour intégrer les deux clients dans une seule application, il faut procéder la manière suivante :

- ◇ Récupérer les fichiers sources du chargeur d'applications et du client de sécurité;
- ◇ Supprimer le répertoire *HeavyClient* contenu dans les sources du client de sécurité;
- ◇ Copier les répertoires *GUI*, *HeavyClient*, *RequestManager*, *ToolsBox* et le fichier *language.properties* depuis les sources du projet du client d'applications vers les sources du client de sécurité;
- ◇ Ouvrir le fichier *ApplicationManager.java* et supprimer la méthode *main()*;
- ◇ Ouvrir le fichier *ClientConf.java*;
- ◇ Modifier la valeur de la constante *PATH* par le répertoire où sera installé le client lourd au niveau de l'appareil mobile;
- ◇ Modifier la constante *LOCAL_EXECUTION* à *false*;
- ◇ Compiler le projet et créer l'archive Jar.

Au niveau des certificats, il est nécessaire d'importer le certificat du serveur d'applications dans le *truststore* existant de l'appareil mobile afin que les applications du client lourd puissent communiquer avec leur serveur. Il est également nécessaire d'importer le certificat du serveur de sécurité dans le *truststore* utilisé par le serveur d'applications pour les contrôles des identifiants de sécurité des applications. Pour de plus amples informations concernant la méthode à suivre pour importer les certificats, voir le chapitre 19.6.2 à la page 113.

Il est maintenant nécessaire d'ajouter les applications sur le serveur de sécurité afin que celles-ci soient automatiquement ajoutées sur les clients lourds :

- ◇ Ouvrir le service d'administration à l'aide d'un navigateur Internet;
- ◇ Ajouter les applications *Alert*, *FichePatient*, *RendezVous* en tant qu'applications non prioritaires, mais installables automatiquement¹⁹;
- ◇ Ajouter l'application *System* en tant qu'application prioritaire et installable automatiquement;
- ◇ Ajouter les fichiers suivants dans les applications :
 - *ahooga.wav* et *NativeToolsBox* dans *System*;
 - *Alert.jar* et *FichePatient.jar* dans *Alert*;

19 En cas de problèmes pour l'ajout des applications et des fichiers, se référer au chapitre 19.3.10, page 105.

- *FichePatient.jar* dans *FichePatient*;
- *RendezVous.jar* dans *RendezVous*;
- ◇ Ajouter toutes les applications ci-dessus dans la liste des applications que le groupe de l'utilisateur a le droit d'utiliser;
- ◇ Remplacer le fichier *Client.jar* sur le serveur de sécurité par la version nouvellement compilée;

Au niveau du serveur d'applications, il est nécessaire de lui préciser d'accéder au serveur de sécurité pour contrôler les identifiants et non à une méthode locale retournant une valeur simulée :

- ◇ Ouvrir les sources du serveur d'applications (*Serveur_TD*);
- ◇ Ouvrir la classe *LogiqueMetier.java* présente dans le paquetage *servlet*;
- ◇ Modifier la constante *LOCAL_EXECUTION* à *false*, puis recompiler et relancer le serveur;

L'intégration est maintenant terminée. Le service devrait fonctionner correctement.

Remarque :

Les tests effectués sur le client de sécurité dans le cadre de ce projet ont été réalisés avec l'application de simulation des requêtes. Il n'a en revanche pas été possible d'effectuer de tests approfondis sur l'ensemble du service regroupant le service de sécurité et le service d'applications, car ces deux parties sont menées à être modifiées jusqu'à la fin du projet.

Le guide d'intégration des services est donc donné à titre d'information seulement.

19.5 Installation des outils de développement

Ce guide explique comment installer les outils de développement pour Java ME. Ces fichiers sont également présents sur le CD du projet, dans le répertoire *Development tools*.

19.5.1 Développement sur la plateforme Java Mobile ou Sony Ericsson

1. Télécharger et installer J2SE Development Kit 6
<http://java.sun.com/javase/downloads/index.jsp>
2. Télécharger NetBeans IDE 5.5 et l'installer
 - Sans Java EE :
<http://www.netbeans.info/downloads/index.php>
 - Avec Java EE 5 (sélectionner « Download with tools ») :
<http://java.sun.com/javaee/downloads/index.jsp>
3. Télécharger et installer NetBeans Mobility Pack 5.5 for CDC
<http://www.netbeans.info/downloads/index.php?rs=11&p=10>
4. Télécharger et installer la plateforme de développement UIQ3SDK pour appareils de type Sony/ Ericsson (SDK utilisé pour les librairies AWT) (garder toutes les options par défaut)
<http://developer.sonyericsson.com/getDocument.do?docId=77006>
5. Télécharger et installer P990 Extension Package for the UIQ 3 SDK Beta 3
<http://developer.sonyericsson.com/getDocument.do?docId=78484>
6. Ouvrir le programme SDK Configurator de UIQ3SDK, puis sélectionner le « device » par défaut, cliquer OK. Aller dans l'onglet Styles, sélectionner le premier style (P990 Flip Closed). Cliquer sur Apply, puis quitter le programme.
7. Ouvrir NetBeans, aller dans Tools/Java Platform Manager, cliquer sur Add Platform et sélectionner Java Micro Edition CDC Platform Emulator. Ouvrir ensuite le répertoire où se trouve UIQ3SDK et cliquer sur Next, puis Finish.

19.5.2 Développement pour une plateforme Windows Mobile

1. Suivre le point 19.5.1
2. Télécharger et installer Virtual Machine Network Driver for Microsoft Device Emulator (netsvwrap.msi)
<http://www.microsoft.com/downloads/details.aspx?familyid=DC8332D6-565F-4A57-BE8C-1D4718D3AF65&displaylang=en>
3. Télécharger un émulateur de plateforme Windows Mobile (V1Emulator.zip et efp.msi)
<http://www.microsoft.com/downloads/details.aspx?FamilyId=C62D54A5-183A-4A1E-A7E2-CC500ED1F19A&displaylang=en>
4. Télécharger et installer WebSphere Everyplace Micro Environment 6.1.1, CDC 1.1, Foundation 1.1, Personal 1.1 pour Windows Mobile 5.0
http://www-128.ibm.com/developerworks/websphere/zones/wireless/weme_eval_runtimes.html
5. Télécharger et installer Microsoft Active Sync 4.5 (setup.msi)
<http://www.microsoft.com/downloads/details.aspx?displaylang=fr&FamilyID=9E641C34-6F7F-404D-A04B-DC09F8141141>
6. Ouvrir ActiveSync, aller dans Fichier/Paramètres de connexion, cocher « Autoriser la connexion

à l'un des éléments suivants » et sélectionner DMA.

7. Aller dans le menu démarrer, puis Microsoft Windows Mobile 5.0 MSFP Emulator Images et exécuter Pocket PC Coldboot. Une fois démarré, exécuter Device Emulator Manager, sélectionner le seul appareil disponible et choisir «cradle » pour démarrer la synchronisation.
8. Aller dans le répertoire « C:\Program Files\IBM\WEME\runtimes\61\wm50-arm-ppro11\ », décompresser le fichier « weme-wince50-arm-ppro11_6.1.1.20061110-161633.zip » et copier les répertoires « bin » et « lib » dans « \J9\PPRO11 » sur le Pocket PC.
9. Télécharger et décompresser RAPI Tools (itsutilsbin-20060821.zip)
<http://www.xs4all.nl/~itsme/projects/xda/tools.html>
10. Ouvrir NetBeans, créer un nouveau projet pour CDC et choisir SEMC 100 version 9.00 ...
11. Dans le projet, remplacer le fichier build.xml par le fichier décrit sur le site http://blogs.sun.com/bblog/entry/developing_java_applications_for_pocket entre les 2 balises <project> à la suite de ce qu'il y a déjà dans le fichier.

12. Remplacer la ligne

```
<arg value=""-jcl:ppro10\ -classpath \"${pocketpc.java.install.path}\${dist.jar.name};${pocketpc.java.install.path}\${matisse.layout.jar}\" $
{main.class}\"/>
```

par

```
<arg value="-jcl:ppro11 -classpath ${pocketpc.java.install.path}\${dist.jar.name};$
{pocketpc.java.install.path}\${matisse.layout.jar} ${main.class}\"/>
```

13. Remplacer les différents chemins en fonction de l'emplacement du RAPI Tools et du répertoire J9 sur le Pocket PC, ainsi que le nom du projet.
14. Exécuter le projet.

19.6 Création des certificats

Ce guide explique comment créer les différents certificats nécessaires à la sécurisation du service.

19.6.1 Au niveau du serveur Web

La création d'une clé publique et d'une clé privée au niveau du serveur Web (Apache-Tomcat dans notre cas) est relativement simple :

1. Lancer l'application suivante avec les paramètres suivants :

```
%JAVA_HOME%\bin\keytool.exe -genkey -alias nomService -keyalg RSA
```

où *nomService* est le nom souhaité pour la paire de clés (privée et publique).

2. Saisir les informations demandées; Il faut noter que le *Common Name* doit être identique au nom du serveur utilisé dans l'URL.
3. Un fichier *.keystore* présent dans le profil de l'utilisateur courant sous Windows est alors créé, contenant la clé privée et la clé publique qui sera utilisée par le serveur Web;
4. Ouvrir le fichier *server.xml* dans le répertoire *conf* de Tomcat et remplacer le commentaire

```
<!--  
<Connector port="8443"  
    maxThreads="150" minSpareThreads="25" maxSpareThreads="75"  
    enableLookups="false" disableUploadTimeout="true"  
    acceptCount="100" debug="0" scheme="https" secure="true"  
    clientAuth="false" sslProtocol="TLS" URIEncoding="UTF-8"/>  
-->
```

par

```
<Connector  
    port="8443" minSpareThreads="5" maxSpareThreads="75"  
    enableLookups="true" disableUploadTimeout="true"  
    acceptCount="100" maxThreads="200"  
    scheme="https" secure="true" SSLEnabled="true"  
    keystoreFile="${user.home}/.keystore" keystorePass="security"  
    clientAuth="false" sslProtocol="TLS"/>
```

puis redémarrer le serveur Web.

Il est à noter quand dans notre cas, le fichier *.keystore* est situé dans le répertoire de l'utilisateur local. Il est bien sûr possible de déplacer ce fichier, il faut alors modifier le chemin d'accès *keystoreFile=...*. De plus, le mot de passe pour ouvrir le fichier *.keystore* doit être indiqué dans *keystorePass=...*, il vaut ici *security*.

Le serveur Web est maintenant prêt à recevoir des requêtes sécurisées.

19.6.2 Importation des certificats sur Java ME

Sans ajout du certificat du serveur sur le client mobile, ce dernier refusera d'ouvrir la connexion et

affichera l'erreur suivante : *certificate chain root is not a trusted Certificate Authority*. Cela signifie que le client Java ME ne possède aucune information sur l'autorité de certification qui lui permette de contrôler la validité du certificat du serveur. Il faut donc importer le certificat du serveur sur le client.

Dans un premier temps, il faut extraire la clé publique du serveur sous forme de certificat. Pour cela, il faut effectuer la commande suivante :

```
%JAVA_HOME%\bin\keytool.exe -export -alias nomService -file certFile
```

où *nomService* est le nom du service utilisé précédemment sur le serveur. L'option *-file* permet de spécifier le nom du certificat généré. Cette commande prend par défaut le *keystore* présent dans le répertoire du profil de l'utilisateur courant.

La suite de l'exportation doit s'effectuer à l'aide du programme *keytool* adapté à la machine virtuelle utilisée. Dans mon cas, il s'agit du *keytool* d'IBM fourni dans la suite d'outils nommée « IBM WebSphere Studio Device Developer ». Celle-ci doit être téléchargée sur le site d'IBM directement. Ces fichiers sont également présents dans le répertoire *IBM Files* du CD du projet.

Une fois la suite d'outils installée, il faut ouvrir le dossier ...*\IBM\DeviceDeveloper\wsdd5.0\ive-2.2* (ou *IBM Files* en cas d'utilisation du CD) puis copier le contenu du dossier *lib* dans *%JAVA_HOME%\lib*.

En utilisant une console, aller dans le répertoire ...*\IBM\DeviceDeveloper\wsdd5.0\ive-2.2\bin* (ou *IBM Files\bin*), exécuter la commande suivante et saisir un mot de passe pour le magasin de clés (par exemple « security ») :

```
<Keytool_IBM>\bin\keytool -import -alias nomService -keystore truststore -file certFile
```

où le nom du service doit correspondre à celui utilisé par le serveur et où le nom donné par l'option *-file* doit correspondre à celui utilisé dans la commande précédente. Enfin, le nom donné à l'option *-keystore* est celui qui sera utilisé dans le code Java pour ouvrir le magasin des certificats autorisés.

Le fichier généré (ici *truststore*) doit être copié dans l'appareil mobile. Au niveau du code, il faut ajouter les lignes suivantes avant l'ouverture de la connexion sécurisée :

```
System.setProperty("javax.net.ssl.trustStore", RepTruststore);  
System.setProperty("javax.net.ssl.trustStorePassword", PassTruststore);
```

où *RepTruststore* est le répertoire dans lequel le magasin est stocké (incluant le nom du magasin) et *PassTruststore* est le mot de passe du magasin saisi à la commande précédente.

L'appareil mobile peut maintenant ouvrir une connexion sécurisée sur le serveur.

19.6.3 Exportation du certificat pour le serveur de fichiers

Une exportation du certificat est également nécessaire pour que le serveur de sécurité puisse se connecter au serveur de fichiers de manière sécurisée.

La procédure pour exporter le certificat est très similaire à celle décrite pour les appareils mobiles. Il faut toutefois utiliser le *keytool* Java standard pour créer le *truststore* :

```
%JAVA_HOME%\bin\keytool -import -alias nomService -keystore truststore -file certFile
```

où le nom du service doit correspondre à celui utilisé par le serveur et où le nom donné par l'option *-file* doit correspondre au certificat précédemment exporté. Enfin, le nom donné à l'option *-keystore* est celui qui sera utilisé dans le code Java pour ouvrir le magasin des certificats autorisés.

19.7 Installation du service

Ce guide a pour but d'expliquer comment installer le service de sécurité, tant au niveau du serveur qu'au niveau du client, afin qu'il puisse être rapidement fonctionnel. Les fichiers indiqués dans ce manuel sont contenus dans le répertoire d'installation présent sur le CD du projet.

L'installation du service est prévue pour le système d'exploitation Windows XP Service Pack 2, bien qu'il soit possible d'installer ce service sur d'autres systèmes d'exploitation. Les commandes et les fichiers utilisés risquent alors de différer quelque peu. Il est à noter que le répertoire d'installation présent sur le CD du projet est nommé <ReplInstall> dans ce guide afin de faciliter la lecture de celui-ci.

Remarque :

Le service de sécurité a été créé pour fonctionner sur un serveur nommé *security-server*. Dans le cas où un autre nom de serveur est utilisé, il est nécessaire de modifier l'adresse de connexion du client lourd et de le recompiler.

19.7.1 Installation du serveur

19.7.1.1 Installation de la base de données

- ◇ Décompresser l'archive `mysql-5.0.45-win32.zip` présente dans le répertoire <ReplInstall>\Database Server;
- ◇ Exécuter le fichier *Setup.exe* et effectuer une installation complète;
- ◇ A la fin de l'installation, choisir l'option de configuration du serveur MySQL;
- ◇ Choisir la configuration détaillée, puis choisir le type *Server Machine*;
- ◇ Choisir le type de base de données *Multifuntional Database*;
- ◇ Continuer l'installation avec les paramètres proposés par défaut en s'assurant que le serveur est installé comme service Windows s'exécutant automatiquement au démarrage de ce dernier.
- ◇ Lorsque le mot de passe est demandé, saisir un mot de passe qui sera utilisé pour se connecter à la base de données. Le mot de passe utilisé dans le cadre de ce guide est « security_DB ».

19.7.1.2 Importation de la base de données

- ◇ Pour importer la base de données du service de sécurité, il faut tout d'abord installer un programme qui permette d'effectuer cette tâche. Le programme utilisé ici se nomme *SQLyog*. Installer le fichier *sqlyog_6.05.exe* présent dans le répertoire <ReplInstall>\Database Server;
- ◇ Démarrer *SQLyog*;
- ◇ Se connecter au serveur de base de données avec le nom d'utilisateur « root » et le mot de passe « security_DB »;
- ◇ Importer le fichier SQL *securityDatabase.sql* présent dans le répertoire <ReplInstall>\Database Server. Ce fichier SQL contient la structure de la base de données de sécurité ainsi que les paramètres par défaut du serveur.

19.7.1.3 Installation du serveur Web

Le serveur Web utilisé est le serveur Apache Tomcat incluant le serveur de fichiers WebDAV. L'installation de Java, non détaillée dans ce guide, doit être effectuée avant l'installation du serveur Web.

- ◇ Décompresser l'archive *jakarta-slide-2.1-tomcat-5.0.28.zip* présente dans le répertoire `<RepInstall>\Web Server`;
- ◇ Déplacer le dossier décompressé dans le répertoire de son choix, en prenant garde au fait que les espaces contenus dans le chemin d'accès vers le serveur empêchent son démarrage. Ce répertoire sera nommé `<RepServeur>` pour la suite de ce guide.
- ◇ Démarrer le serveur Web en exécutant `<RepServeur>\bin\startup.bat`;

19.7.1.4 Configuration du serveur de fichiers

Il est nécessaire de configurer le serveur de fichiers afin de lui spécifier des droits d'accès. Pour ce faire, le programme Java *DAVExplorer* est utilisé.

- ◇ Décompresser le fichier *DAVExplorer-latest.zip* présent dans le répertoire `<RepInstall>\Web Server`;
- ◇ Exécuter le fichier *DAVExplorer.bat*;
- ◇ Se connecter au serveur à l'aide de l'adresse `localhost:8080/slide` et en spécifiant le nom de l'utilisateur *root* et le mot de passe *root*;
- ◇ Se rendre dans le dossier *users* au niveau du serveur de fichiers;
- ◇ Supprimer tous les utilisateurs à l'exception de l'utilisateur *root*;
- ◇ Sélectionner l'utilisateur *root*, puis dans l'onglet *View*, sélectionner *View/Modify Properties*;
- ◇ Sélectionner le champ *password* et modifier la valeur *root* par le mot de passe souhaité. Dans le cadre de ce projet, le mot de passe est *security*;
- ◇ Sauvegarder les changements;

19.7.1.5 Configuration du serveur Web

- ◇ Si le serveur Web est encore en cours d'exécution, fermer le serveur Web;
- ◇ Ouvrir le répertoire `<RepServeur>\conf\tomcat-users.xml`;
- ◇ Remplacer le contenu de ce fichier par le contenu suivant (où *password* permet d'assigner le mot de passe souhaité) :

```
<?xml version='1.0' encoding='utf-8'?>
<tomcat-users>
  <role rolename="manager"/>
  <user username="admin" password="securityManager" roles="manager"/>
</tomcat-users>
```

- ◇ Démarrer le serveur Web;
- ◇ Ouvrir un navigateur Internet et se connecter au serveur par l'adresse `http://localhost:8080/`;
- ◇ Cliquer sur le lien *Tomcat Manager* sur la page d'accueil et s'authentifier en donnant le nom

d'utilisateur *admin* et le mot de passe *securityManager*;

- ◇ Supprimer les applications *jsp-examples*, *servlets-examples*, *slide-doc* et *tomcat-docs*;
- ◇ Ajouter l'application serveur *Server.war* présente dans le dossier *<ReplInstall>\Web Server*;
- ◇ Si le mot de passe du serveur de fichiers ou le mot de passe de la base de données ne sont pas identiques à ceux utilisés dans ce guide, il est nécessaire de modifier les paramètres du serveur. Pour plus d'informations, se référer au chapitre 19.3.11, page 107.
- ◇ Contrôler que le service d'administration fonctionne en ouvrant le lien *http://localhost:8080/Server*;

19.7.1.6 Sécurisation du serveur

Cette étape consiste à créer un certificat pour le serveur afin de sécuriser les connexions vers celui-ci.

- ◇ Si le serveur Web est en cours d'exécution, le fermer;
- ◇ Créer les certificats pour le serveur. Les détails concernant la création de ces certificats sont donnés au chapitre 19.6, page 113. Le fichier *SSLConnector.txt* présent dans le répertoire *<ReplInstall>\Web Server* contient le connecteur SSL qu'il est possible d'utiliser pour le fichier *server.xml* du serveur présent dans le répertoire *conf*. Si le mot de passe du magasin de clés a été modifié, il faut aussi modifier ce mot de passe dans le fichier *server.xml*.
- ◇ Modifier le fichier *configuration.xml* présent dans le dossier *\webapps\Server\WEB-INF* du serveur si nécessaire pour modifier les chemins vers les certificats (certificat du serveur Web et *truststore* vers le serveur de fichiers).
- ◇ Démarrer le serveur Web et se connecter au lien *https://localhost:8443/Server* afin de contrôler que le chiffrement par SSL fonctionne correctement.

19.7.1.7 Configuration de base du serveur

Le serveur, tel qu'il est configuré, n'est pas encore utilisable. Afin que le serveur soit pleinement fonctionnel, il est nécessaire de lui ajouter l'application du client lourd, de lui paramétrer des accès pour les utilisateurs et pour les appareils mobiles et enfin de lui ajouter des paramètres de sécurité. Pour ce faire, il faut utiliser le service d'administration :

- ◇ Accéder au service d'administration : *https://localhost:8443/Server*;
- ◇ Se connecter avec le nom *admin* et le mot de passe *admin*;
- ◇ Créer une application *Client* contenant les fichiers *client.jar* et *mobileDeviceServices.dll* présents dans le répertoire *<ReplInstall>\Client* (chapitre 19.3.10, la page 105);
- ◇ Ajouter la machine virtuelle utilisée par les appareils mobiles (chapitre 19.3.8, page 104);
- ◇ Ajouter le système d'exploitation utilisé par les appareils mobiles en spécifiant l'application *Client* comme client lourd et en spécifiant la machine virtuelle créée précédemment (chapitre 19.3.9, page 104);
- ◇ Ajouter un nouvel appareil mobile (chapitre 19.3.6, page 103);
- ◇ Ajouter un nouvel ensemble de paramètres de sécurité (chapitre 19.3.5, page 102);
- ◇ Ajouter un nouveau groupe d'utilisateurs (chapitre 19.3.4, page 102);
- ◇ Ajouter un nouvel utilisateur (chapitre 19.3.3, page 101);

- ◇ Assigner l'appareil mobile à cet utilisateur (chapitre 19.3.7, page 103);

Le serveur de sécurité possède à présent les paramètres minimaux pour être pleinement fonctionnel.

19.7.2 Installation du client lourd

L'installation du client lourd est relativement simple. Il est en revanche nécessaire de procéder à des configurations de Windows Mobile et de l'appareil mobile afin que le client lourd puisse être utilisé dans des conditions optimales.

19.7.2.1 Configuration de Windows Mobile

Dans un premier temps, il est nécessaire de corriger un problème de gestion de mémoire de Windows Mobile. En effet, Windows Mobile « gère » lui-même l'attribution de la mémoire aux applications et ne ferme ainsi pas les applications lorsqu'elles devraient être fermées, dans le but théorique de pouvoir les ouvrir à nouveau plus rapidement si l'utilisateur le désire. Malheureusement, cette gestion ne fonctionne pas et les machines virtuelles fermées restent en mémoire jusqu'à ce que la mémoire sature et empêche l'ouverture d'une quelconque application. Une application tierce, nommée *smclose.exe*, doit être installée afin de forcer Windows mobile à libérer la mémoire lors de la fermeture des applications [25].

- ◇ Copier le fichier *smclose.exe* présent dans le répertoire *<RepInstall>\Patches Windows Mobile* dans le répertoire *\Windows\Démarriage* afin que ce fichier soit exécuté à chaque démarrage;
- ◇ Exécuter ce fichier sur l'appareil mobile (une seule fois);

Il est par ailleurs conseillé de désactiver la mise en veille automatique du PDA, car celle-ci peut entraîner des coupures de réseaux pour le client lourd. Pour la désactiver, aller dans le menu *Démarrer\Paramètres\Système\Alimentation\Avancé* et de désactiver la mise en veille automatique.

19.7.2.2 Installation de la machine virtuelle

Dans un deuxième temps, il est nécessaire d'installer la machine virtuelle sur l'appareil mobile. L'installation de cette dernière est extrêmement simple :

- ◇ Aller dans le répertoire *<RepInstall>\Virtual Machine* et copier le répertoire J9 à la racine de l'appareil mobile;

Il est à noter que la librairie SWT est directement incluse dans la machine virtuelle du CD du projet, il n'est donc pas nécessaire de l'installer séparément.

19.7.2.3 Installation des certificats

Il est nécessaire d'ajouter le certificat du serveur dans la racine du PDA afin que ce dernier puisse communiquer avec le serveur de manière sécurisée. Ce certificat doit être nommé *truststore* et son mot de passe doit être *security*. En cas d'utilisation d'un autre nom ou d'un autre mot de passe pour le *truststore*, il faut modifier les constantes adéquates au niveau du client lourd et le recompiler.

Pour plus d'informations concernant la création du certificat, se reporter au chapitre 19.6 à la page 113.

Remarque :

Si des applications du client lourd nécessitent des connexions sécurisées vers d'autres serveurs, il est nécessaire d'effectuer l'importation des certificats dans le *truststore* pour chaque serveur.

19.7.2.4 Installation du client lourd

En fonction des applications qui seront ajoutées sur le client lourd, ce dernier peut requérir un espace en mémoire assez important. Il est de ce fait recommandé de placer le client lourd sur une carte mémoire afin de ne pas saturer la mémoire de l'appareil mobile.

Le répertoire dans lequel le client lourd est placé sera nommé <RepClient> dans ce guide.

- ◇ Copier le fichier *Client.jar* du répertoire <ReplInstall>\Client dans le répertoire <RepClient> du PDA;
- ◇ Créer un raccourci pour ouvrir ce fichier. Un modèle de raccourcis est donné ci-dessous :

```
255#"J9\PPRO11\bin\j9.exe" -jcl:ppro11 -classpath "<RepClient>\Client.jar"  
client.ClientUpdateManager
```

Il suffit de modifier le répertoire <RepClient> par le chemin absolu vers le fichier, puis d'enregistrer cette ligne de texte dans un fichier dont l'extension sera « .lnk ».

- ◇ Placer ce raccourci dans le répertoire <RepClient>, puis l'exécuter. Lors de la première exécution du client lourd, plusieurs messages d'avertissements apparaissent demandant si les bibliothèques sont autorisées à être exécutées par le système d'exploitation. Accepter chacun de ces messages.

19.7.2.5 Connexion à un réseau Wi-Fi protégé par WPA2 et TKIP

Windows Mobile 5.0 gère relativement mal l'installation des certificats pour des réseaux sans fil d'entreprise. Nous avons utilisé un logiciel tiers pour installer notre certificat afin de pouvoir se connecter au réseau. Les étapes principales suivantes sont extraites d'une marche à suivre en ligne [26] et sont rappelées ci-dessous :

- ◇ Télécharger le logiciel *pfximprt* du site <http://www.jacco2.dds.nl/networking/pfximprt.html#Download> ;
- ◇ Extraire l'archive, puis copier le fichier *pfximprt.exe* sur le PDA. Ce programme permet d'importer des certificats sur Windows Mobile 5.0. Ces certificats doivent cependant être enregistrés au format *pfx*.
- ◇ Exporter le certificat personnel permettant de se connecter au réseau de l'entreprise, puis le convertir au format *pfx*. Nous nommerons ce fichier *certPerso.pfx*.
- ◇ Exporter le certificat du ou des CA (*Certificate Authority*) ayant émis le certificat personnel. L'exportation doit se faire au format DER X.509 (.cer). Dans notre cas, nous le nommerons *certifCA.cer*.
- ◇ Copier ces deux certificats sur l'appareil mobile (ou plus si d'autres CA sont présents);
- ◇ Sur l'appareil mobile, ouvrir le certificat *certifCA.cer* afin de l'installer; Faire de même pour les autres certificats de CA si nécessaire;
- ◇ Exécuter le programme *pfximprt.exe* sur le PDA, puis importer le certificat personnel *certPerso.pfx*;
- ◇ Enclencher le Wi-Fi, puis cliquer sur le réseau sans fil désiré et cliquer sur *Connexion*. Dans la page de paramètres qui s'ouvre, choisir *Internet*, sélectionner le cryptage *TKIP* et le type EAP : *Carte à puce ou certificat*, puis cliquer sur *Terminer*. Une fenêtre risque de s'ouvrir demandant le nom de l'utilisateur et le domaine, cliquer sur *OK* jusqu'à ce que la fenêtre n'apparaisse plus.
- ◇ Windows Mobile 5 devrait à présent être apte à se connecter au réseau sans fil.

19.8 Journal de travail

Ce journal de travail est basé sur un modèle créé par l'Ecole Technique et des Métiers de Lausanne (ETML) et représente les durées planifiées et les durées effectives pour chaque tâche du projet.

19.8.1 Résumé

Cette partie résume la durée planifiée et la durée effective pour chaque tâche.

Journal de travail

Nom du projet :	Mobile Doctor's Desktop – service de sécurité et de mise à jour pour clients mobiles
Prénom - nom :	Stephen Badan
Classe :	TR 2007
Lieu :	Yverdon-les-Bains, Route de Cheseaux
Dates :	Du 18.09.2997 au 12.12.2007
Durée totale [Périodes] :	59
Durée d'une période [Jours] :	1

Planification des activités du module :

Tâches - objectifs	Durée planifiée [Jours]	Durée effective [Jours]	Durée effective [h]	Notions, activités, délais, priorités, remarques
Absences - imprévus	0	5	40	
Echanges de fichiers XML	2	2	16	
Mise en place base de données	2	1	8	
Implémentation connexion SSL	3	5	40	
Implémentation authentification formelle	1	2	16	
Implémentation ID temporaire de sécurité	1	2	16	
Implémentation authentification matérielle	5	3	24	
Implémentation contrôle des applications	5	3	24	
Transfert fichiers entre client/serveur	3	1	8	
Stockage et chargement fichiers sur client	3		0	
Gestion mises à jour	5	10	80	
Connexion Web sécurisée	2		0	
Page authentification administrateur	1	1	8	
Gestion du contenu de la base de données	7	7	56	
Envoi applications vers le serveur pour le service d'administration	4	1	8	
Tests et corrections	5	14	112	
Implémentation de l'authentification externe	0	2	16	
Rapport	10	19	152	
Impression et rendu du rapport	5	5	40	
Total	64	83	0	

19.8.2 Détails

La partie ci-dessous donne les détails de l'avancement des tâches par semaine, en commençant par la première semaine.

Date :	Du 04.09.2007 au 07.09.2007		
Tâche	Temps effectué [Périodes]	Explications: qu'est-ce qui a été fait et comment ?	Suite à donner, références, liens, documents reçus, ...
Echanges de fichiers XML	2	Implémentation des communications en XML entre le client et le serveur en utilisant JAXB et kXML	
Implémentation authentification formelle	1	Implémentation de la boîte de dialogue sur le client et des contrôles sur le serveur	
Implémentation connexion SSL	2	Implémentation d'une connexion sécurisée à l'aide de TLS entre le client et le serveur en utilisant un certificat du côté du serveur	
Total session	5		

Date :	Du 11.09.2007 au 14.09.2007		
Tâche	Temps effectué [Périodes]	Explications: qu'est-ce qui a été fait et comment ?	Suite à donner, références, liens, documents reçus, ...
Implémentation connexion SSL	1	Implémentation d'une connexion sécurisée à l'aide de TLS entre le client et le serveur en utilisant un certificat du côté du serveur	
Implémentation ID temporaire de sécurité	2	Implémentation d'un identificateur de sécurité pour que le serveur reconnaisse le client une fois authentifié	
Implémentation authentification formelle	1	Transformation de l'interface graphique en utilisant SWT au lieu de AWT en raison de limitations d'AWT	
Implémentation contrôle des applications	1	Début de l'implémentation des échanges de messages entre le client et le serveur dans le but de faire des contrôles de sécurité d'applications	En raison d'un programme manquant, j'ai commencé cette étape avant d'effectuer l'étape de l'authentification matérielle
Total session	5		

Date :	Du 18.09.2007 au 21.09.2007		
Tâche	Temps effectué [Périodes]	Explications: qu'est-ce qui a été fait et comment ?	Suite à donner, références, liens, documents reçus, ...
Implémentation authentification matérielle	3	Recherche et implémentation d'une méthode permettant d'extraire le code IMEI du téléphone et de le vérifier sur le serveur	
Rapport	1	Mise au propre du journal de travail et début du rapport	
Absences - imprévus	1	Jeûne fédéral	
Total session	5		

Date :	Du 25.09.2007 au 28.09.2007		
Tâche	Temps effectué [Périodes]	Explications: qu'est-ce qui a été fait et comment ?	Suite à donner, références, liens, documents reçus, ...
Implémentation contrôle des applications	2	Implémentation de la gestion des authentifications et création d'applications d'exemple pour tester le bon fonctionnement de la gestion des authentifications	
Implémentation contrôle des applications	1	Création d'une application d'exemple	
Tests et corrections	1	Tests du service de gestion des demandes d'authentification	
Tests et corrections	1	Résolution d'un problème de fermeture lié à l'application de test. L'erreur "aléatoire" empêchant parfois certaines threads de se terminer à la fermeture de l'application était due à l'écriture dans une zone de texte de la fenêtre pendant la fermeture de la fenêtre, situation créant un blocage.	
Total session	5		

Date :	Du 01.10.2007 au 06.10.2007		
Tâche	Temps effectué [Périodes]	Explications: qu'est-ce qui a été fait et comment ?	Suite à donner, références, liens, documents reçus, ...
Transfert fichiers entre client/serveur	1	Mise en place d'un serveur Webdav pour stocker les fichiers, implémentations des méthodes nécessaires à récupérer des fichiers incluant des contrôles de droits des utilisateurs.	
Stockage et chargement fichiers sur client	1	Correction d'un problème lié à la transmission par SSL (invalid padding) en mettant en place une transmission des fichiers par paquets multiples de 8 bytes. De plus, début du stockage des applications sur le client mobile.	
Gestion mises à jour	1	Implémentation du contrôle de version de la machine virtuelle.	
Gestion mises à jour	1	Implémentation du contrôle de la version du client lourd	
Gestion mises à jour	1	Implémentation des contrôles des applications prioritaires	
Rapport	1	Rapport sur les services d'authentifications	
Total session	6		

Date :	Du 08.10.2007 au 12.10.2007		
Tâche	Temps effectué [Périodes]	Explications: qu'est-ce qui a été fait et comment ?	Suite à donner, références, liens, documents reçus, ...
Gestion mises à jour	1	Ajout fenêtre de choix des mises à jour non-prioritaires et gestion de ces mises à jour	
Gestion mises à jour	1	Fin du service de mise à jour et premiers tests	
Gestion mises à jour	1	Ajout de contrôles des codes hachés des fichiers et ajout d'une fenêtre de bienvenue	
Gestion mises à jour	1	Ajout de la possibilité de mettre à jour des applications non-jar et tests approfondis	
Tests et corrections	1	Tests du service de mise à jour et correction des erreurs rencontrées	
Gestion mises à jour	1	Ajout de la possibilité de mettre à jour des applications contenant plusieurs fichiers.	
Total session	6		

Date :	Du 15.10.2007 au 20.10.2007		
Tâche	Temps effectué [Périodes]	Explications: qu'est-ce qui a été fait et comment ?	Suite à donner, références, liens, documents reçus, ...
Gestion mises à jour	2	Fin des mises à jour d'applications ayant des fichiers multiples	
Page authentification administrateur	1	Mise en place de la structure du site et mise en place de l'authentification de l'administrateur	
Gestion du contenu de la base de données	2	Début de l'ajout des pages de saisies pour les tables de la base de données	
Tests et corrections	1	Tests et ajout d'un second identifiant de sécurité pour les applications	
Rapport	1	Rédaction du rapport sur la sécurité	
Total session	7		

Date :	Du 22.10.2007 au 26.10.2007		
Tâche	Temps effectué [Périodes]	Explications: qu'est-ce qui a été fait et comment ?	Suite à donner, références, liens, documents reçus, ...
Gestion du contenu de la base de données	3	Suite de l'ajout des pages de saisies pour les tables de la base de données	
Envoi applications vers le serveur pour le service d'administration	1	Implémentation de l'envoi des fichiers vers le serveur et gestion du stockage des fichiers sur le serveur de fichier et des informations sur les fichiers dans la base de données	
Gestion du contenu de la base de données	1	Fin de l'implémentation de la gestion du contenu de la base de données	
Rapport	1	Rédaction du rapport sur les mises à jour	
Total session	6		

Date :	Du 29.10.2007 au 03.11.2007		
Tâche	Temps effectué [Périodes]	Explications: qu'est-ce qui a été fait et comment ?	Suite à donner, références, liens, documents reçus, ...
Tests et corrections	1	Contrôle et corrections afin que le site de gestion du serveur soit conforme à XHTML 1.0 transitional	
Tests et corrections	1	Corrections multiples dans le code au niveau des termes utilisés et tests de l'ensemble du service	
Gestion du contenu de la base de données	1	Externalisation de la configuration du serveur dans un fichier XML et dans la base de données	
Tests et corrections	1	Ajout de la fonctionnalité d'authentification externe à l'aide d'un autre appareil bluetooth à proximité	
Rapport	2	Corrections de la partie du service des authentifications dans le rapport et rédaction du fonctionnement du service de mise à jour	
Total session	6		

Date :	Du 05.11.2007 au 09.11.2007		
Tâche	Temps effectué [Périodes]	Explications: qu'est-ce qui a été fait et comment ?	Suite à donner, références, liens, documents reçus, ...
Rapport	1	Fin du rapport sur les mises à jour et tests divers	
Tests et corrections	1	Corrections et amélioration du processus de fermeture du client lourd	
Absences - imprévus	1	Forum HES-SO'07 à Martigny	
Tests et corrections	1	Tests et corrections de problèmes liés au chiffrement SSL (invalid padding)	
Tests et corrections	1	Fin des tests du service de mise à jour	
Tests et corrections	1	Correction d'un problème lié à l'envoi de fichiers volumineux sur le client mobile.	
Total session	6		

Date :	Du 12.11.2007 au 17.11.2007		
Tâche	Temps effectué [Périodes]	Explications: qu'est-ce qui a été fait et comment ?	Suite à donner, références, liens, documents reçus, ...
Tests et corrections	1	Contrôle des échanges d'ID entre les serveurs	
Tests et corrections	1	Tests du service d'administration	
Rapport	1	Corrections et mises à jour des schémas blocs du service dans le rapport	
Rapport	1	Rédaction du rapport concernant les technologies utilisées	
Rapport	1	Fin du rapport concernant le manuel d'utilisation du service d'administration	
Rapport	1	Fin du rapport concernant le manuel d'utilisation du client lourd	
Total session	6		

Date :	Du 19.11.2007 au 24.11.2007		
Tâche	Temps effectué [Périodes]	Explications: qu'est-ce qui a été fait et comment ?	Suite à donner, références, liens, documents reçus, ...
Tests et corrections	1	Installation du serveur de sécurité sur une machine fixe	
Tests et corrections	1	Intégration du chargeur d'applications dans le client de sécurité et tests	
Absences - imprévus	1	Réunion à ELCA Lausanne	
Rapport	1	Explications des diagrammes des classes et manuel d'installation du serveur et du client	
Rapport	1	Rédaction du guide d'intégration des services du client lourd	
Total session	5		

Date :	Du 26.11.2007 au 30.11.2007		
Tâche	Temps effectué [Périodes]	Explications: qu'est-ce qui a été fait et comment ?	Suite à donner, références, liens, documents reçus, ...
Tests et corrections	1	Tests finaux du service	
Rapport	4	Contrôle et suite de rédaction du rapport	
Total session	5		

Date :	Du 03.12.2007 au 07.12.2007		
Tâche	Temps effectué [Périodes]	Explications: qu'est-ce qui a été fait et comment ?	Suite à donner, références, liens, documents reçus, ...
Rapport	1	Corrections du rapport	
Absences - imprévus	2	Recrutement militaire	
Rapport	2	Finalisation du rapport	
Total session	5		

Date :	Du 10.12.2007 au 14.12.2007		
Tâche	Temps effectué [Périodes]	Explications: qu'est-ce qui a été fait et comment ?	Suite à donner, références, liens, documents reçus, ...
Impression et rendu du rapport	5	Impression et rendu du rapport et du support numérique (CD) (durée approximative)	
Total session	5		

20. Lexique

Les abréviations importantes utilisées dans ce document sont rappelées ci-dessous.

Abréviation	Signification
API	Application Programming Interface
AWT	Abstract Windowing Toolkit
CA	Certificate Authority
CDC	Connected Device Profile
CLDC	Connected Limited Device Configuration
CSS	Cascading Style Sheets
DLL	Dynamic Link Library
GSM	Global System for Mobile Communications
HTML	HyperText Markup Language
JAAS	Java Authentication and Authorization Service
Java EE	Java Enterprise Edition
Java ME	Java Micro Edition
Java SE	Java Standard Edition
JAXB	Java Architecture for XML Binding
JNI	Java Native Interface
JSP	Java Server Pages
MAC	Medium Access Control
MVC	Modèle – Vue – Contrôleur
PDA	Personal Digital Assistant
SGBD	Système de Gestion de Base de Données
SGML	Standard Generalized Markup Language
SIM	Subscriber Identity Module
SSL	Secure Sockets Layers
SSO	Single Sign-On
SWT	Standard Widget Toolkit
TLS	Transport Layer Security
WebDAV	Web-based Distributed Authoring and Versioning
XML	eXtensible Markup Language

21. Index des figures

Figure 1: Diagramme de contexte du service de sécurité.....	12
Figure 2: Schéma général du service.....	15
Figure 3: Structure détaillée du service.....	17
Figure 4: Structure détaillée du service de sécurité.....	19
Figure 5: Profils Java.....	20
Figure 6: Classification des méthodes d'authentification biométrique.....	25
Figure 7: Diagramme temporel de la gestion des authentifications en situation normale.....	34
Figure 8: Diagramme temporel de la gestion des authentifications en situation d'évitement.....	35
Figure 9: Fenêtre d'authentification formelle.....	35
Figure 10: Authentification matérielle sur Windows Mobile.....	37
Figure 11: Représentation en couches de l'authentification matérielle.....	38
Figure 12: Fenêtre d'authentification matérielle.....	39
Figure 13: Exemple d'empreinte digitale utilisée lors d'une l'authentification biométrique.....	40
Figure 14: Authentification externe à l'aide de la technologie Bluetooth.....	41
Figure 15: Exemple d'applications pour le service de mise à jour.....	42
Figure 16: Mise à jour d'une application prioritaire.....	46
Figure 17: Fenêtre de sélection des mises à jour.....	48
Figure 18: Diagramme d'activités des mises à jour et de l'installation des applications.....	49
Figure 19: Représentation graphique du schéma XSD.....	51
Figure 20: Génération des identifiants de sécurité.....	56
Figure 21: Création et échange des identifiants de sécurité.....	57
Figure 22: Fenêtre principale d'authentification.....	58
Figure 23: Structuration du site Internet d'administration.....	60
Figure 24: Schéma conceptuel de la base de données.....	61
Figure 25: Schéma conceptuel de la base de données.....	63
Figure 26: Diagramme des classes du client.....	67
Figure 27: Diagramme des classes du serveur.....	70
Figure 28: Macro-calendrier réel.....	71
Figure 29: Macro-calendrier initial.....	71
Figure 30: Graphique des durées de chaque tâche.....	72
Figure 31: Micro-calendrier – détails.....	73
Figure 32: Micro-calendrier – général.....	73
Figure 33: Démarrage du client lourd.....	92
Figure 34: Page d'accueil.....	92
Figure 35: Mise à jour du client lourd.....	93
Figure 36: Fin de la mise à jour du client lourd.....	93
Figure 37: Avertissement de redémarrage du client lourd.....	93
Figure 38: Mise à jour de la librairie d'authentification.....	93
Figure 39: Authentification matérielle.....	94
Figure 40: Liste de choix des utilisateurs.....	94
Figure 41: Fenêtre d'authentification formelle.....	94
Figure 42: Erreur de Bluetooth lors de l'authentification externe.....	95
Figure 43: Authentification externe par Bluetooth.....	95
Figure 44: Fenêtre de choix des mises à jour.....	95
Figure 45: Transfert d'une application non prioritaire.....	96
Figure 46: Annulation d'un téléchargement.....	96
Figure 47: Contrôle des fichiers communs à plusieurs applications.....	97

Figure 48: Application de test.....	97
Figure 49: Résultats des requêtes simulées.....	98
Figure 50: Identifiant de sécurité invalide.....	99
Figure 51: Compte de l'utilisateur bloqué.....	99
Figure 52: Connexion sur le site d'administration.....	100
Figure 53: Représentation du menu des tâches.....	100
Figure 54: Gestion des utilisateurs.....	101
Figure 55: Attribution des appareils mobiles.....	104
Figure 56: Ajout et suppression de fichiers dans une application.....	106

22. Index lexical

A	
API.....	20, 21, 124
Authentification.....	
Biométrique.....	13, 18, 25, 28-31, 34, 35, 40, 41, 74
Externe.....	8, 11, 13, 18, 24, 25, 27-30, 32, 34, 40, 41, 61, 62, 66, 71, 74, 75, 78-80, 94, 101-103
Formelle.....	8, 10, 11, 13, 18, 24-35, 41, 47, 55, 58, 62, 66, 79, 80, 94, 99, 102, 103
Matérielle.....	8, 10, 13, 18, 24-26, 28-31, 34, 36-38, 41, 47, 66, 78, 81, 84, 93, 94
AWT.....	21, 111, 124
B	
Base de données.....	4, 11, 18, 23, 44, 61, 63, 68, 69, 79, 84, 88, 108, 115, 117, 124
C	
CA.....	53, 119, 124
CDC.....	20, 21, 36, 38, 111, 112, 124
Certificat.....	24, 36, 53, 108, 109, 113, 114, 117-119, 124
Classe.....	11, 18, 38, 39, 50, 60, 65, 66, 68, 69, 109, 110
CLDC.....	20, 124
Code haché.....	37, 38, 44, 47, 55, 62, 69, 74, 84, 88, 106
Contrôleur.....	18, 69, 124
CSS.....	60, 124
D	
Délai.....	27, 29, 32-35, 43, 44, 55, 62, 63, 66, 77-81, 85, 94, 99, 102, 103, 106, 107
DLL.....	38, 39, 82, 124
G	
GSM.....	36, 124
H	
HTML.....	22, 60, 124
I	
IBM.....	21, 22, 41, 78, 104, 112, 114
IMEI.....	36-39, 41, 54, 55, 61, 65, 66, 69, 86, 103
Intégrité.....	8, 12, 16, 37, 42-44, 62, 65, 74, 75, 93, 95, 106
Interface graphique.....	21, 65, 66, 79, 91
Invalid padding.....	53, 71
J	
J9.....	21, 41, 78, 104, 112, 118, 119
JAAS.....	40, 124
Java EE.....	22, 111, 124
Java ME.....	20-22, 36, 38, 50, 113, 114, 124
Java SE.....	20-22, 38, 124
JAXB.....	11, 22, 50, 68, 124
JNI.....	21, 22, 36, 39, 41, 124
JSP.....	22, 23, 58, 59, 69, 124
M	
MAC.....	41, 75, 80, 101, 124
Machine virtuelle.....	21, 36, 38, 41, 46, 47, 53, 58, 62, 78, 81, 84, 87, 91, 98, 104, 105, 114, 117, 118
Modèle.....	4, 18, 37, 39, 47, 61, 63, 69, 119, 120, 124
MVC.....	18, 124
MySQL.....	23, 115

<i>P</i>	
PDA.....	4, 20, 38, 41, 74-76, 91, 118, 119, 124
<i>S</i>	
Servlet.....	22, 23, 50, 54, 58, 68, 69, 110, 117
Seuil.....	28
SIM.....	24, 36, 38, 124
SSL.....	10, 11, 14, 52, 53, 71, 89, 117, 124
SSO.....	40, 124
SWT.....	22, 118, 124
Symbian.....	21, 40
<i>T</i>	
TLS.....	14, 52, 53, 113, 124
<i>U</i>	
Use case.....	12
<i>V</i>	
Vue.....	18, 39, 58, 62, 71, 77, 124
<i>W</i>	
WebDAV.....	23, 69, 91, 116, 124
Windows Mobile.....	21, 22, 36-40, 46, 75, 78, 92, 105, 111, 112, 118, 119
<i>X</i>	
XML.....	10, 11, 22, 50, 51, 54, 65, 68, 109, 124
.	
.NET.....	21

23. Liste des références

- 1 : Gene Spafford's Personal Pages: Quotable Spaf, *auteur* : E. H. Spafford , *lien* : <http://homes.cerias.purdue.edu/%7Espaf/quotes.html>
- 2 : Java ME Technology, *auteur* : Sun Microsystems , *lien* : <http://java.sun.com/javame/technology/index.jsp>
- 3 : J2ME grows up, *auteur* : Sun Microsystems , *lien* : <http://www.ibm.com/developerworks/java/library/j-j2me/>
- 4 : Personal Profile, *auteur* : Sun Microsystems , *lien* : <http://java.sun.com/products/personalprofile/>
- 5 : IBM WebSphere Everyplace Micro Environment for Windows Mobile Personal Profile 1.1, *auteur* : , *lien* : <http://www.handango.com/ampp/store/PlatformProductDetail.jsp?siteId=1827&jid=845XBB6XEB86F96293ED2FC2313X19E7&platformId=2&productType=2&catalog=0&sectionId=0&productId=204403>
- 6 : Site officiel de SWT, *auteur* : The Eclipse Foundation , *lien* : <http://www.eclipse.org/swt/>
- 7 : Standard Widget Toolkit, *auteur* : Wikipedia , *lien* : http://en.wikipedia.org/wiki/Standard_Widget_Toolkit
- 8 : XML, *auteur* : Wikipedia , *lien* : <http://en.wikipedia.org/wiki/Xml>
- 9 : Caractéristiques officielles de MySQL, *auteur* : MySQL AB , *lien* : [MySQL AB](http://www.mysql.com/doc/tech-resources/charsets/mysql.html)
- 10 : Jakarta Slide, *auteur* : Wikipedia , *lien* : http://en.wikipedia.org/wiki/Jakarta_Slide
- 11 : WebDAV, *auteur* : Wikipedia , *lien* : <http://en.wikipedia.org/wiki/WebDAV>
- 12 : Netcraft, *auteur* : NETCRAFT LTD , *lien* : <http://news.netcraft.com/>
- 13 : Apache Tomcat, *auteur* : Wikipedia , *lien* : http://en.wikipedia.org/wiki/Apache_tomcat
- 14 : Biométrie et Authentification, *auteur* : Djamil Mahmoudi , *lien* : <http://ditwww.epfl.ch/SIC/SA/publications/FI00/fi-sp-00/sp-00-page25.html>
- 15 : GSME proposals regarding mobile theft and IMEI security, *auteur* : The European interest group of the GSM Association , *lien* : http://www.gsmworld.com/gsm europe/documents/positions/gsme_proposals_mobile_thefts_imei_security.pdf
- 16 : Windows Mobile 5.0 SDK for Pocket PC, *auteur* : Microsoft , *lien* : <https://www.microsoft.com/downloads/details.aspx?familyid=83A52AF2-F524-4EC5-9155-717CBE5D25ED&displaylang=en>
- 17 : Fingerprint SDK Biometric Recognition Library, *auteur* : Griaule , *lien* : http://www.griaule.com/page/en-us/grfinger_fingerprint_sdk
- 18 : Building Biometric Authentication for J2EE, Web, and Enterprise Applications, *auteur* : Sun Microsystems , *lien* : <http://developers.sun.com/prodtech/identserver/reference/techart/bioauthentication.html>
- 19 : Advanced Biometric Controls, *auteur* : Advanced Biometric Controls , *lien* : <http://www.biobex.com/main.htm>
- 20 : Parsing XML in J2ME, *auteur* : Sun Microsystems , *lien* : <http://developers.sun.com/techtopics/mobility/midp/articles/parsingxml/>
- 21 : Eclipse Public Licence, *auteur* : Wikipedia , *lien* : http://en.wikipedia.org/wiki/Eclipse_Public_License#_note-eplfaq
- 22 : Transport Layer Security, *auteur* : Wikipedia , *lien* : http://en.wikipedia.org/wiki/Transport_Layer_Security#Fonctionnement
- 23 : The Sietch Blog (source pour image), *auteur* : - , *lien* : <http://www.blog.thesietch.org/category/tools/>
- 24 : Download free CSS templates, *auteur* : freecsstemplates.org , *lien* : <http://www.freecsstemplates.org/>
- 25 : PocketPCFreeware - smclose, *auteur* : PocketPcFreeware , *lien* : <http://www.pocketpcfreewares.com/fr/index.php?soft=1453>
- 26 : Freephonie.org, *auteur* : Freephonie.org , *lien* : <http://www.freephonie.org/doku/certificatswm5>